



Facultad 4

**Sistema de gestión para la asignación y control de los recursos materiales
en la Facultad 4**

Trabajo de diploma para optar por el título Ingeniero en Ciencias Informáticas

Autor: Daniela Muñoz Molina

Tutor(es): Ing. Maydalis Hernández Pérez

Ing. Alina D. Rodríguez Peña

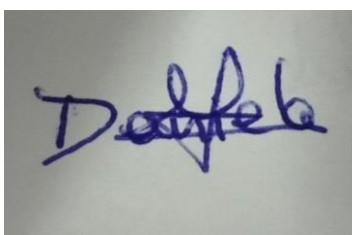
La Habana, diciembre de 2021

“Año 63 de la Revolución”

DECLARACIÓN DE AUTORÍA

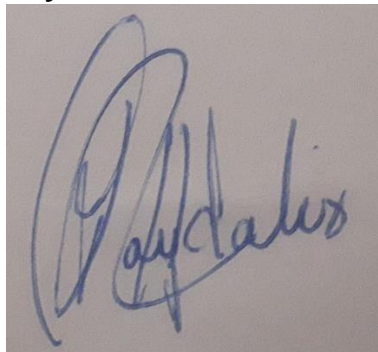
Declaro por este medio que yo, Daniela Muñoz Molina, soy la autora del trabajo de diploma con título “Sistema de gestión para la asignación y control de los recursos materiales en la Facultad 4” y concedo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la investigación, con carácter exclusivo. Para que así conste, firmo la presente declaración de autoría el día 1 del mes de diciembre del año 2021.

Daniela Muñoz Molina



Firma del Autor

Maydalis Hernández Pérez



Alina D. Rodríguez Peña



Firma del Tutor

Firma del Tutor

Datos de contacto

DATOS DE CONTACTO

Maydalis Hernández Pérez, Ingeniera en Ciencias Informáticas. Profesora en el departamento de informática de la Universidad de Ciencias Informáticas. Correo electrónico: mhernandep@uci.cu

Alina D. Rodríguez Peña, Ingeniera en Ciencias Informáticas. Vicedecana de Administración y Economía de la Facultad 4. Correo electrónico: alina@uci.cu

AGRADECIMIENTOS

Quiero agradecer en un principio a mi familia por su incondicional apoyo en todo lo que he tenido que enfrentar y en todo lo que me he propuesto hacer. Por el amor que siempre me han brindado, por ayudarme a crecer y convertirme en una mejor persona; simplemente gracias por ser mi familia.

También, agradecer a mi familia UCI, por su gran apoyo durante estos años en la UCI. Cada uno con su peculiar manera de ser, de una forma u otra me enseñaron algo. Gracias por ello, por su amistad incondicional, confianza, preocupación y dedicación.

Gracias a mis tutoras por todo su apoyo, especialmente a Maydalis por ser mi guía durante todo este proceso y por siempre dedicar un pedacito de su tiempo a atenderme y ayudarme. Gracias a todos mis profesores desde primero a quinto año por haber aportado un granito de arena en mi formación durante todo mi periodo de estudiante.

A mis compañeros de aula y de apartamentos. A todos aquellos que compartieron gratos momentos conmigo y que ya no se encuentran en la UCI.

¡¡¡¡¡A todos mis más sinceras GRACIAS!!!!!!

Dedicatoria

DEDICATORIA

A toda mi familia y amigos debido a que, sin ellos, su comprensión, confianza, consejos y apoyo no lo hubiera logrado.

A todos lo que han hecho posible que hoy me gradué como ingeniera en ciencias informáticas.

A ustedes con todo mi corazón.

RESUMEN

Con el constante desarrollo de las Tecnologías de la Información y las Comunicaciones, surgen nuevos sistemas informáticos que facilitan el control y gestión administrativo en las empresas. La presente investigación tiene como objetivo el diseño de un sistema de gestión para la asignación y control de los recursos materiales en la facultad 4. Para dar cumplimiento al objetivo planteado se definieron los principales conceptos asociados a la temática y se realizó una investigación a sistemas similares. Además, se generaron los principales artefactos del flujo de trabajo propuestos por la metodología de desarrollo *Agile Unified Process* (AUP) en su versión UCI. También, se describen las herramientas necesarias para el desarrollo de la propuesta de solución, así como las posibles pruebas y técnicas a realizar para verificar y validar el correcto funcionamiento del sistema.

PALABRAS CLAVE: control, asignación, sistema administrativo.

ABSTRACT

With the constant development of Information and Communications Technologies (ICT), new computer systems are emerging that facilitate the control and administrative management in companies. The objective of this research is to development of a management system for the assignment and control of material resources in Faculty 4. In order to comply with the proposed objective, the main concepts associated with the topic were defined and an investigation was carried out on similar systems. In addition, the main workflow artifacts proposed by the Agile Unified Process (AUP) development methodology in its UCI version were generated. It also describes the necessary tools to develop the solution proposal, as well as the possible tests and techniques to carry out to verify and validate the correct operation of the system.

KEYWORDS: control, assignment, management system.

TABLA DE CONTENIDOS

INTRODUCCIÓN	1
CAPÍTULO I: FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS SOBRE EL 6	6
OBJETO DE ESTUDIO	6
1.1 Procesos de negocio	6
1.2 Análisis de los procesos de negocio vinculados a la asignación y el control de los recursos materiales en la Facultad 4.....	7
1.2.1 Análisis de los procesos Asignación y Control de recursos	15
1.3 Sistemas similares.....	16
1.3.1 Análisis del estudio de los sistemas similares	17
1.4 Metodologías de desarrollo de software, lenguajes y herramientas para el modelado	18
1.4.1 Metodología de desarrollo de software	19
1.4.2 Lenguaje de modelado	20
1.4.3 Herramienta de modelado	21
1.5 Lenguajes y Tecnologías de programación	22
1.5.1 Lenguajes de programación	22
1.5.2 Frameworks de desarrollo	24
1.5.3 Entorno de desarrollo	26
1.6 Sistemas Gestores de Base de Datos	28
1.7 Conclusiones del capítulo	29
CAPÍTULO II: DISEÑO DE LA SOLUCIÓN PROPUESTA AL PROBLEMA CIENTÍFICO	31
2.1 Disciplina de modelado del negocio.....	31
2.2 Propuesta de solución	33
2.3 Disciplina de requisitos.....	33
2.3.1 Requisitos Funcionales	34
2.3.2 Descripción de requisitos	41
2.3.3 Requisitos no funcionales	44
2.4 Arquitectura del software.....	45
2.4.1 Diagrama de clases del diseño	46
2.5 Patrones de diseño.....	47
2.6 Modelo de datos	48
2.6.1 Descripción de las tablas	49
2.7 Conclusiones del capítulo	49
CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBA DEL SISTEMA	50
3.1 Disciplina de implementación.....	50
3.2 Estándares de codificación	52
3.3 Pruebas de Software	53
3.3.1 Nivel de unidad	54
3.3.2 Nivel de integración	57
3.3.3 Nivel del sistema	57
3.3.4 Nivel de aceptación	59
3.3.5 Pruebas de regresión y confirmación.....	59
3.4 Validación	60
3.5 Conclusiones del capítulo	61
CONCLUSIONES FINALES	63

RECOMENDACIONES.....	63
REFERENCIAS BIBLIOGRÁFICAS	64
ANEXOS	67

ÍNDICE DE TABLAS

Tabla 1: Comparación de los lenguajes de programación Python, PHP y JavaScript [Tomado de: (27)]	23
Tabla 2: Comparación de los frameworks de Python	25
Tabla 3: Comparación de los IDE de Python [Tomado de: (34)]	27
Tabla 4: Comparación de Sistemas Gestores de Base de Datos [Tomado de: (39)]	29
Tabla 5: Requisitos Funcionales.	34
Tabla 6. DRP.1 Crear tarjeta de préstamos	41
Tabla 7: Descripción de Libro.	49
Tabla 8. CP1 Adicionar Libro (Elaboración propia).	59
Tabla 9: Roles y Responsabilidades (Elaboración propia).	68
Tabla 10: Descripción de Estudiante (Elaboración propia).	74
Tabla 11: Descripción de Trabajador (Elaboración propia).	74
Tabla 12: Descripción de Usuario (Elaboración propia).	74
Tabla 13: Descripción de Personal (Elaboración propia).	75
Tabla 14: Descripción de Recursos Materiales (Elaboración propia).	75
Tabla 15: Descripción del Tipo de material (Elaboración propia).	75
Tabla 16: Descripción de Tarjeta de préstamos (Elaboración propia).	76
Tabla 17: Descripción de la relación de Libro y Tarjeta de préstamos (Elaboración propia).	76
Tabla 18: Descripción de Módulo (Elaboración propia).	76
Tabla 19: Descripción de Módulo por asignatura (Elaboración propia).	77
Tabla 20: Descripción del Módulo por año académico (Elaboración propia).	77
Tabla 21: Descripción de la relación de Recursos Materiales y Módulo (Elaboración propia).	77
Tabla 22: Descripción de la relación de Recursos Materiales y Estudiante (Elaboración propia).	77
Tabla 23: Descripción de la relación de Recursos Materiales y Trabajador (Elaboración propia).	78
Tabla 24: Descripción de Rol (Elaboración propia).	78
Tabla 25: CP2 Eliminar Libro (Elaboración propia).	79
Tabla 26: CP3 Modificar datos del libro (Elaboración propia).	79
Tabla 27: CP4 Ver datos de un libro (Elaboración propia).	80

ÍNDICE DE FIGURAS

Ilustración 1. Estructura jerárquica de la Facultad teniendo en cuenta los roles que interviene en los procesos de Asignación y Control de los recursos materiales en la Facultad 4 (Elaboración propia)....	7
Ilustración 2: Descripción del proceso: Asignación de recursos (Elaboración propia).....	8
Ilustración 3. Descripción del subproceso: Solicitar Materiales (Elaboración propia).	9
Ilustración 4. Descripción del subproceso: Entrega recursos materiales (Elaboración propia).	9
Ilustración 5: Descripción del subproceso: Entrega de insumos de limpieza (Elaboración propia).	10
Ilustración 6: Descripción del subproceso: Entregar libros a estudiantes o trabajadores (Elaboración propia).	11
Ilustración 7: Descripción del proceso: Recepcionar libros (Elaboración propia).	12
Ilustración 8: Descripción del subproceso: Recepcionar libros entregados a estudiantes (Elaboración propia).	13
Ilustración 9: Descripción del subproceso: Recepcionar libros entregados a los trabajadores (Elaboración propia).	14
Ilustración 10: Descripción del subproceso: Controlar los libros en el almacén (Elaboración propia). .	15
Ilustración 11: Descripción del subproceso: Control de recursos (Elaboración propia).	15
Ilustración 12. Modelo To-Be del proceso Entregar módulo por año académico a estudiantes (Elaboración propia).	32
Ilustración 13: Modelo To-Be del proceso Entregar libro a estudiantes (Elaboración propia).	33
Ilustración 14: Interfaz de DRP 1. Crear Tarjeta de Préstamos de estudiante (Elaboración propia). ...	43
Ilustración 15: Interfaz de DRP 1. Crear Tarjeta de Préstamos de trabajador (Elaboración propia). ...	44
Ilustración 16: Diagrama de Clases del diseño Gestionar Libros (Elaboración propia).	46
Ilustración 17: Modelo Entidad-Relación (Elaboración propia).	49
Ilustración 18: Diagrama de Componente (Elaboración propia).	52
Ilustración 19: Diagrama de Despliegue (Elaboración propia).	53
Ilustración 20: Nodos y caminos a recorrer (57).	56
Ilustración 21: Grafo de flujo (57).	57
Ilustración 22: Caminos independientes (57).	58
Ilustración 23: Modelo To-Be del Proceso Asignación de recursos (Elaboración propia).	69
Ilustración 24: Modelo To-Be del proceso Entregar recursos materiales (Elaboración propia).	70
Ilustración 25: Modelo To-Be del proceso Entregar insumos de limpieza (Elaboración propia).	70
Ilustración 26: Modelo To-Be del proceso Recepcionar libros entregados a los estudiantes (Elaboración propia).	71
Ilustración 27: Modelo To-Be del proceso Controlar libros en el almacén (Elaboración propia).	72
Ilustración 28: Modelo To-Be del proceso Control de recursos (Elaboración propia).	72
Ilustración 29: Diagrama de Clases del diseño Gestionar Grupo (Elaboración propia).	73
Ilustración 30: Diagrama de Clases del diseño Gestionar Estudiante (Elaboración propia).	73

INTRODUCCIÓN

En la actualidad las empresas para cumplir con sus objetivos se apoyan en la utilización adecuada de las Tecnologías de la Información y las Comunicaciones (TIC). Estas constituyen un conjunto de sistemas y productos que captan la información del entorno, la almacenan, la procesan, la comunican y la hacen inteligible a las personas. Además, representan la transmisión de la misma a través de la interconexión de equipos que facilitan la construcción de redes (1). El uso de estas tecnologías ha provocado una importante transformación en todos los niveles de la organización, siendo la actividad de innovación una de las beneficiadas (2).

Actualmente, las TIC son utilizadas en las entidades para analizar los datos generados y elaborar información útil para la toma de decisiones. Además, son empleadas para la promoción de los servicios y productos, y para la búsqueda y selección de sus trabajadores. Es decir, según el sector de pertenencia de la entidad, existen soluciones especializadas que tributan al mejoramiento continuo en cada una de sus áreas, permitiendo optimizar los procesos de negocio y elevando así la eficiencia operacional y la reducción de los costos.

Específicamente en las instituciones educativas las TIC se han convertido en instrumento indispensable para ayudar a enriquecer, transformar y completar la trayectoria académica del estudiantado, a través de softwares interactivos, tabletas, páginas web, video conferencias, correo electrónico, chats, entre otros (3). Pero no solo han sido empleadas para mejorar el proceso de enseñanza y aprendizaje, sino que también son utilizadas para realizar actividades administrativas como: controlar y gestionar la información de los estudiantes, egresados, docentes y administrativos, logrando agilidad, oportunidad, seguridad y calidad en la información. Además, permiten dar seguimiento al estado financiero, recursos materiales y capital humano de estas entidades.

En el caso de la Universidad de las Ciencias Informáticas (UCI) se han desarrollado varios sistemas que tributan a la informatización de diferentes procesos entre ellos: el Sistema de Gestión Universitaria (Akademos) que apoya el proceso de formación de pregrado y posgrado y el Sistema de Gestión Administrativa (Kainos), donde se pueden gestionar actividades relacionadas con Control Interno, Capital Humano y Activos Fijos Tangibles (AFT). Sin em-

bargo; actualmente en las facultades aún existen procesos que no se realizan con la calidad requerida como es: la asignación y control de recursos materiales.

Específicamente en la Facultad 4 de la UCI, desde el Vicedecanato de Administración y Economía, se supervisa el control sobre el cuidado, protección y conservación de los recursos materiales que posee la facultad sobre la base de un adecuado uso en función de las actividades que se desarrollen. Para esto es necesario realizar el control: del módulo docente que se entrega a estudiantes, profesores y alumnos ayudantes, de los libros en uso y los existentes y de los insumos y materiales de limpieza que se entregan. Es objeto de revisión en las múltiples inspecciones que se realizan en la facultad y al Vicedecanato las evidencias de control de los recursos y de destino final de los mismos.

Actualmente existen las siguientes dificultades en el proceso de asignación y control de los recursos materiales:

- El histórico de la cantidad de recursos entregados a la facultad y asignados a los trabajadores y estudiantes se encuentran en archivos impresos por años, lo que dificulta la toma de decisiones y la revisión del destino final.
- La Dirección de Servicios Generales entrega una copia física de la factura de insumos de limpieza que se entregan a la Facultad. El Administrador del Docente le da entrada y salida en el Libro de Control de Insumos (estos se distribuyen a las Auxiliares Generales de Servicio o a las oficinas teniendo en cuenta las necesidades). En este proceso se dificulta el control del destino final.

En el caso del almacén de libros:

- El préstamo a estudiantes y trabajadores se realiza en tarjetas. Uno de los datos que se debe tomar es el título del libro que se presta, pero por errores humanos la legibilidad de la letra o no tomar correctamente el nombre o tomo del libro puede incidir negativamente en el proceso de recogida. Además, las tarjetas de préstamo pueden extraviarse o deteriorarse lo que puede provocar la pérdida de los recursos.
- La gestión de cantidades se realiza de forma manual lo que dificulta la entrega de información, por ejemplo, saber de un título en específico la cantidad total (existencia en almacén + préstamos), conocer cuántos estudiantes de un grupo o un año no han

entregado los libros o cuántos no han solicitado préstamos al almacén requiere un tiempo considerable al encargado de almacén.

Teniendo en cuenta la situación antes descrita se define como **problema de investigación**: ¿Cómo optimizar la gestión de los procesos de asignación y control de los recursos materiales en la Facultad 4?

A partir del problema planteado se identifica como **objeto de estudio**: procesos de asignación y control de recursos en las empresas.

Para dar solución al problema de investigación planteado se define como **objetivo**: diseñar un sistema de gestión para la asignación y control de los recursos materiales en la Facultad 4.

Se delimita dentro del objeto de estudio de la investigación el **campo de acción**: sistemas informáticos para la asignación y control de recursos.

Para dar cumplimiento al objetivo planteado es necesario realizar las siguientes **tareas de investigación**:

- Elaboración del marco teórico de la investigación mediante el estudio de sistemas informáticos que permitan la asignación y control de recursos materiales.
- Análisis de los procesos de negocios relacionados con la asignación y control de los recursos materiales en la Facultad 4.
- Definición de la metodología de desarrollo de software, herramientas y tecnologías a utilizar para el desarrollo del sistema.
- Análisis y diseño del sistema teniendo en cuenta lo establecido en la metodología de desarrollo de software seleccionada.
- Definición de la estrategia de pruebas a realizar al sistema para garantizar su correcto funcionamiento.
- Definición de la estrategia de validación de la solución.

Para la realización de las tareas expuestas anteriormente se emplearon los siguientes **métodos de investigación**:

Métodos teóricos: constituyen el enfoque general para abordar los problemas científicos, permitiendo profundizar en las regularidades esenciales de los fenómenos.

- **Histórico-lógico:** posibilitó la investigación sobre el inicio de la aplicación de las TIC en la gestión de los procesos de negocio de una organización, centrándose en el desarrollo de sistemas que permitan la asignación y control de los recursos materiales. Permitió caracterizar las tecnologías utilizadas en el desarrollo de la solución.
- **Analítico-sintético:** permitió durante el proceso investigativo el estudio, análisis e identificación de los conceptos relacionados con la temática abordada. Posibilitó realizar el análisis teórico e identificar los principales conceptos a incluir en la fundamentación teórica y el análisis de la información. Facilitó el análisis de las herramientas similares que responden al objeto de estudio de la investigación, definiéndose un conjunto de indicadores para realizar una comparación entre las herramientas, con el objetivo de identificar características que tributen al desarrollo de la propuesta.
- **Modelado:** permitió crear abstracciones con el objetivo de explicar la realidad. Es empleado como herramienta para la comprensión del problema a resolver y para la creación de los modelos que permitan el diseño de la solución. Posibilitó, a través de los artefactos que propone la metodología de desarrollo de software seleccionada, la creación de diagramas para el desarrollo de los componentes visuales.

Métodos Empíricos: estudian los fenómenos, objetos y procesos observables, confirmados a través de la hipótesis y la teoría.

- **Entrevista:** se entrevistó a la Vicedecana de Administración de la facultad 4 para poder establecer el alcance del presente trabajo, el objetivo, el flujo actual de la información, así como las reglas del negocio. Dentro de los tipos de entrevistas se seleccionó la semiestructurada, debido a que permite una mayor libertad y flexibilidad en la obtención de información.
- **Observación:** posibilitó estudiar y observar el funcionamiento de sistemas similares, identificándose funcionalidades y características que se tuvieron en cuenta para el desarrollo de la solución.
- **Encuestas:** permitió conocer las opiniones y valoraciones que sobre la temática poseen los encuestados y se utilizó para recolectar datos. Se empleó para validar el resultado con el empleo de la técnica de ladov, que constituye una vía indirecta para el estudio de la satisfacción, debido a que los criterios que se emplean tienen su

fundamentación en las relaciones de tres preguntas a través del cuadro lógico de ladov, intercaladas dentro de un cuestionario y que son desconocidas para el sujeto.

- Revisión documental: permitió la obtención de información relacionada con la temática abordada mediante el asesoramiento a través de diferentes artículos y libros.

El presente trabajo cuenta con la siguiente estructura capitular:

Capítulo 1: Fundamentos y referentes teórico-metodológicos sobre el objeto de estudio. En este capítulo se definen los conceptos necesarios para la comprensión del problema relacionado con la asignación y control de los recursos materiales en la Facultad 4. Se realiza el estudio y definición de la metodología, herramientas y tecnologías utilizadas en el desarrollo de la solución.

Capítulo 2: Diseño de la solución propuesta al problema científico.

Se realiza una descripción de la propuesta de solución y se exponen los requisitos funcionales y no funcionales identificados. Teniendo en cuenta la metodología de desarrollo de software seleccionada se realiza el análisis y diseño del sistema, y se presentan los artefactos ingenieriles que documentan el proceso de desarrollo del software.

Capítulo 3: Implementación y prueba del sistema.

En este capítulo se presentan los elementos relacionados con la implementación del sistema. Se define la estrategia de pruebas a realizar para verificar el cumplimiento de los requisitos de software.

CAPÍTULO I: FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS SOBRE

EL OBJETO DE ESTUDIO

La base teórica del presente trabajo se encuentra sustentada a la descripción de los principales conceptos relacionados con la problemática, al estudio realizado a sistemas similares para poder determinar las características esenciales de las mismas. Además, se describen las herramientas, tecnologías y metodología que se emplearán para el desarrollo del sistema.

1.1 Procesos de negocio

Los procesos de negocio constituyen el conjunto de tareas, actividades y procesos que se realizan en la organización, relacionados entre sí, con la intención de proporcionar servicios y productos. Estos procesos deben ser entendidos, modelados y documentados para un mejor desempeño y logro de los objetivos en la organización.

Atendiendo a la naturaleza del trabajo, la forma de organizarse y la industria, los procesos de negocio pueden clasificarse en (4):

- Procesos operacionales: relacionados con el negocio central y la cadena de valor de la empresa. Son los encargados de generar y entregar al cliente el valor en forma de producto o servicio.
- Procesos de apoyo: constituyen el soporte a los procesos operacionales y le generan valor al cliente de forma directa.
- Procesos de gestión o estratégicos: monitorean y controlan las actividades relacionadas con los procedimientos y sistemas. No crean valor al cliente de forma directa. Ejemplo de esto es la planeación estratégica y la infraestructura del negocio. En el caso específico de los procesos de apoyo el empleo de las TIC trae consigo grandes beneficios como la reducción de tiempos de ciclo y de inventario, la proporción de la información actualizada y fiable y la toma de decisiones (5).

A continuación, se muestra un análisis de algunos procesos considerados como procesos de apoyo en la Facultad 4, los cuales se encuentran relacionados con la asignación y control de los recursos materiales en la facultad 4.

1.2 Análisis de los procesos de negocio vinculados a la asignación y el control de los recursos materiales en la Facultad 4

El Vicedecanato de Administración y Economía, tiene entre sus objetivos asignar, controlar, proteger y conservar los recursos materiales que posee la facultad sobre la base de un adecuado uso en función de las actividades que se desarrollen. En este departamento se definen y se ejecutan dos macroprocesos que son objetos de estudio para esta investigación: la Asignación de recursos y el Control de recursos.

Para el análisis de estos procesos primeramente hay que definir los roles que intervienen en su ejecución y estos son: la Decana de la facultad, el Vicedecano de Administración y Economía, el Asistente de Control, el Administrador del docente, el responsable del almacén, los trabajadores, estudiantes y personal de servicio. En la Ilustración 1, que se muestra a continuación, se realiza un organigrama en el que se representa la estructura jerárquica de los roles que intervienen en la ejecución de estos procesos de negocio en la Facultad 4. La descripción de estos roles se encuentra en el Anexo A.

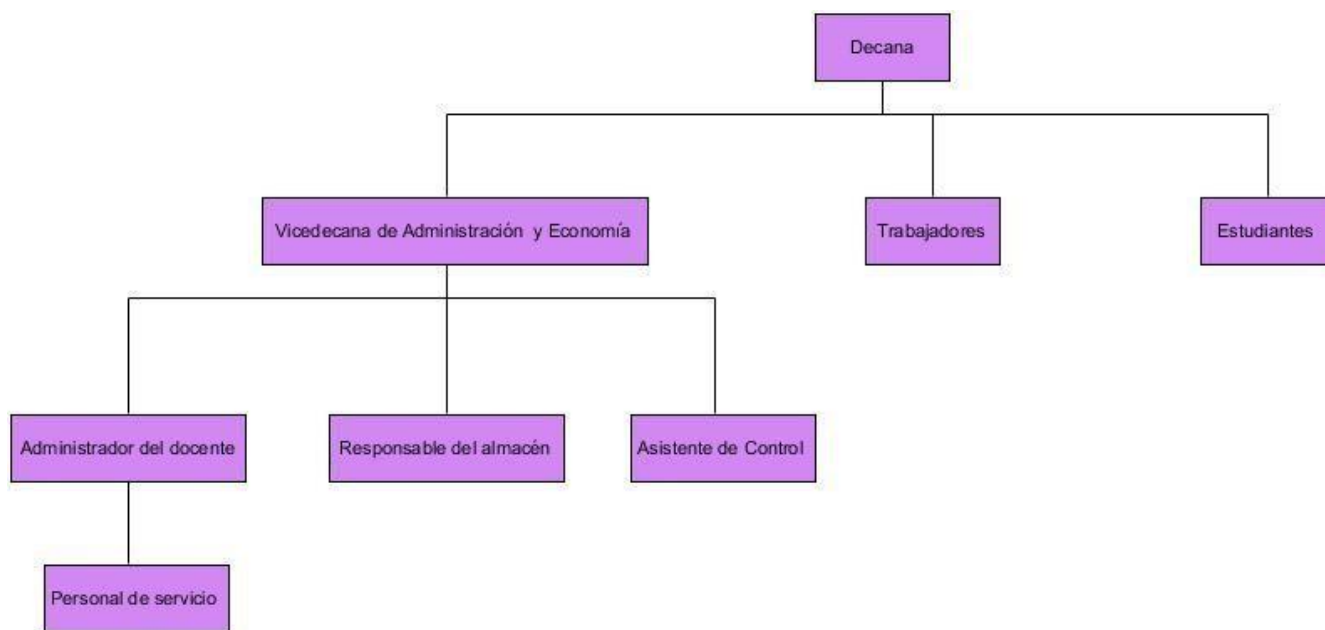


Ilustración 1. Estructura jerárquica de la Facultad teniendo en cuenta los roles que interviene en los procesos de Asignación y Control de los recursos materiales en la Facultad 4 (Elaboración propia).

Una vez explicada la estructura jerárquica se explica el flujo de actividades que integran estos dos procesos:

1. Asignación de recursos

En el proceso de asignación de recursos (ver Ilustración 2) primeramente, se debe verificar si es necesario realizar una solicitud de recursos a la universidad. En caso de ser necesario se debe realizar primero el subproceso Solicitar recursos y posteriormente se puede hacer entrega de diferentes recursos (libros, insumos de limpieza y recursos académicos) que están en correspondencia con la ejecución de cualquier combinación de los siguientes subprocesos: Entregar libros a estudiantes o trabajadores, Entregar insumos de limpieza y Entregar recursos académicos.

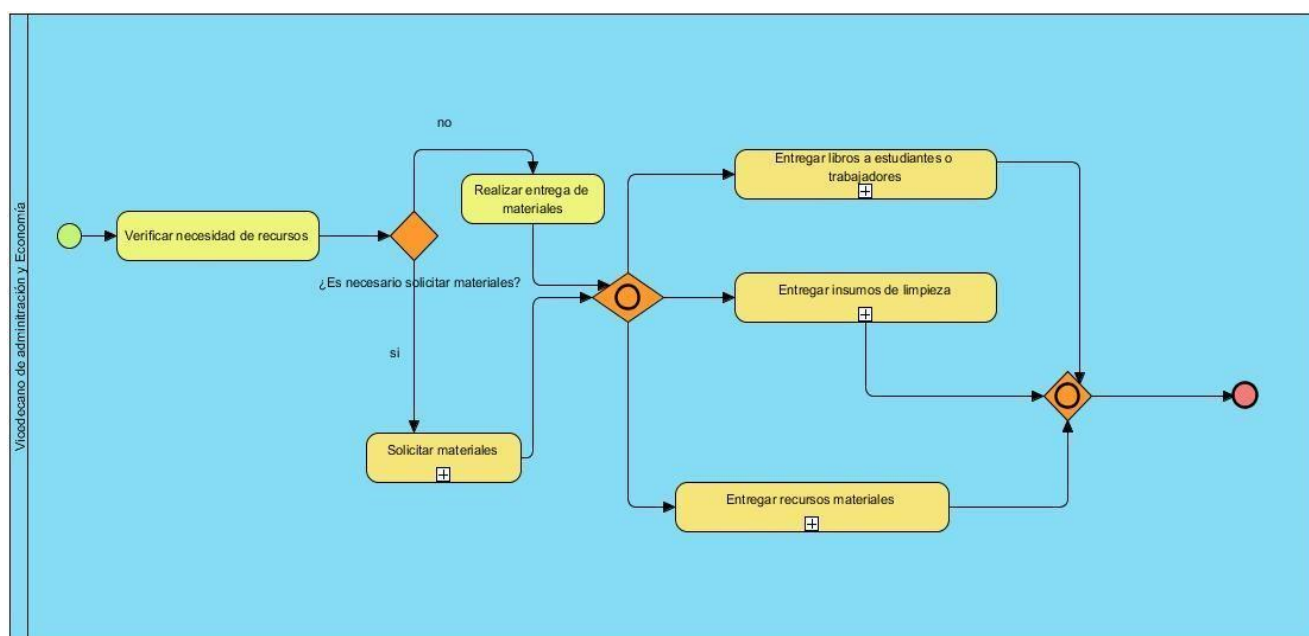


Ilustración 2: Descripción del proceso: Asignación de recursos (Elaboración propia).

A continuación, se describen los subprocesos definidos en el proceso de Asignación de recursos:

• Solicitar materiales

Para la solicitud de materiales (ver Ilustración 3), la Asistente de Control entrega a los directivos responsables de los recursos a nivel de Universidad una Solicitud de Materiales (SM). En caso de ser aprobada, los directivos firman la solicitud entregada y la Asistente de Control procede a la extracción de los recursos del almacén. El administrador del docente, junto a los recursos solicitados, entrega el Vale de Salida (VS) del almacén para dejar constancia de la operación, este VS es archivado en el Vicedecanato de Administración junto a la copia de la SM realizada.

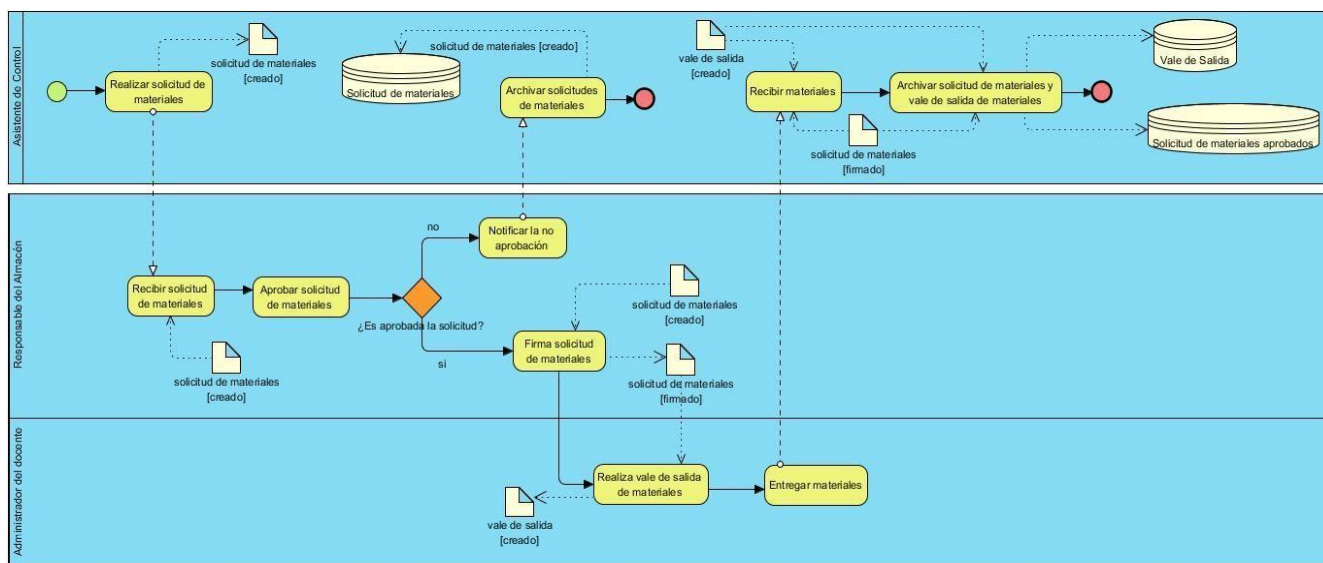


Ilustración 3. Descripción del subproceso: Solicitar Materiales (Elaboración propia).

• Entregar recursos materiales

Los recursos materiales son entregados al solicitante, que pueden ser estudiantes, trabajadores o jefes de departamento. El responsable de entregar estos recursos es el Asistente de Control, el cual debe dejar constancia de la entrega en un acta de destino final que se realiza. Esta acta debe ser debidamente firmada y archivada (ver Ilustración 4).

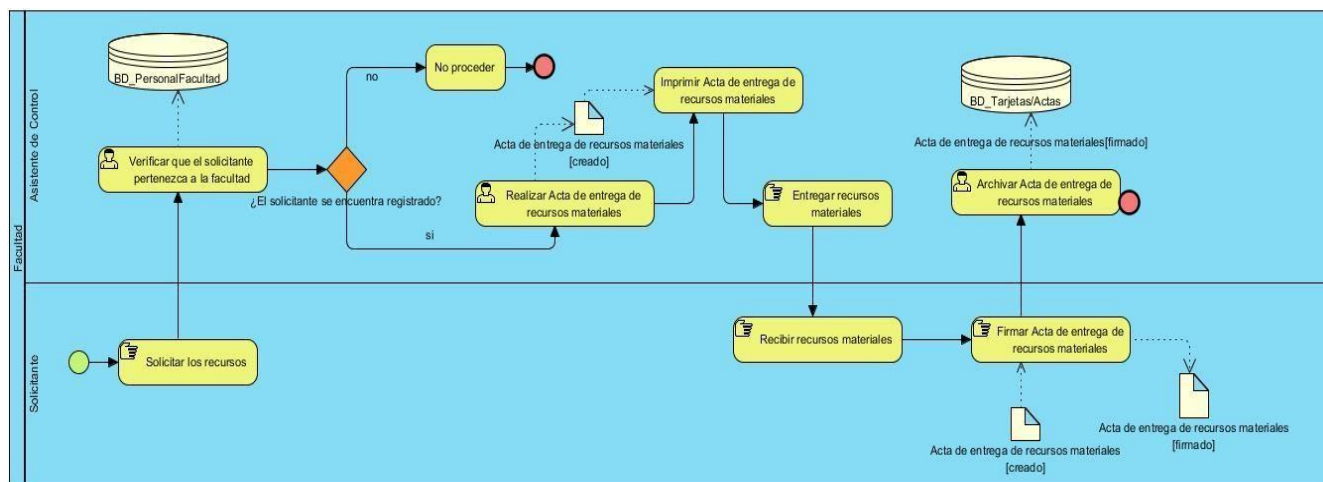


Ilustración 4. Descripción del subproceso: Entrega recursos materiales (Elaboración propia).

• Entregar insumos de limpieza

Cuando los insumos son entregados al administrador (ver Ilustración 5), este debe darle entrada en el Libro de control de insumos. Una vez que se realice esta actividad, el administrador debe entregar los insumos a las Auxiliares Generales de Servicio o a las oficinas

teniendo en cuenta las necesidades. Cuando se realice la entrega se debe firmar un acta como constancia.

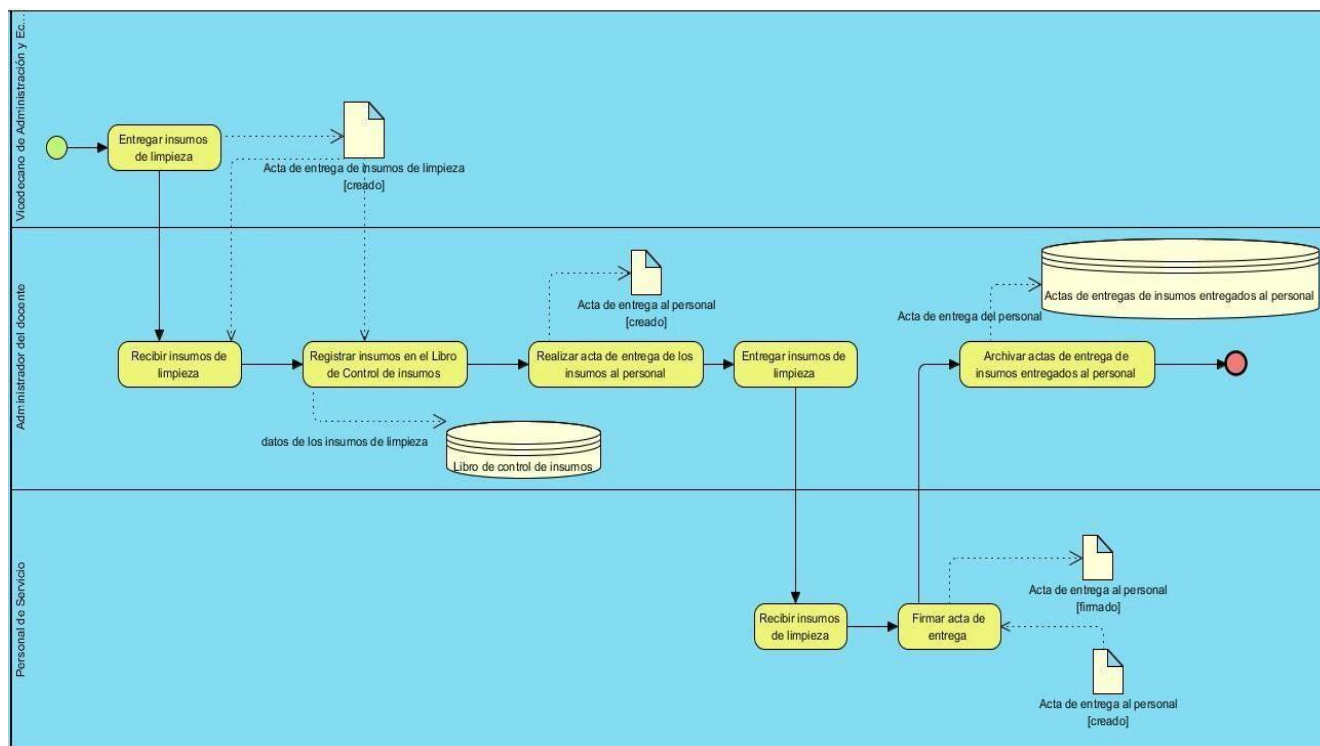


Ilustración 5: Descripción del subproceso: Entrega de insumos de limpieza (Elaboración propia).

2. Entregar libros a estudiantes o trabajadores:

Para la entrega de libros los solicitantes (estudiantes o trabajadores) se presentan en el almacén de libros de la facultad y solicitan los libros que desean (ver Ilustración 6). Una vez que lo solicitan el responsable del almacén debe verificar si la persona pertenece a la facultad o no. En caso de no pertenecer a la facultad se le notifica que no se puede proceder con la entrega de los libros. En caso contrario, se busca si el solicitante tiene creada una tarjeta de préstamo, si ya tiene una tarjeta se actualiza, en caso contrario se crea una en la que se registran los siguientes datos: nombre y apellidos, facultad, año, número de solapín, título de los libros, fecha de entrega, firma de entrega, fecha de recibo, firma de recibo y en caso de ser trabajador se le añade el cargo y el área al que pertenece. Una vez que la responsable del almacén le entrega los libros el solicitante debe firmar la entrega. Estas tarjetas son archivadas en el almacén teniendo en cuenta el año y grupo de los estudiantes, y en caso de ser trabajador el área al que pertenece.

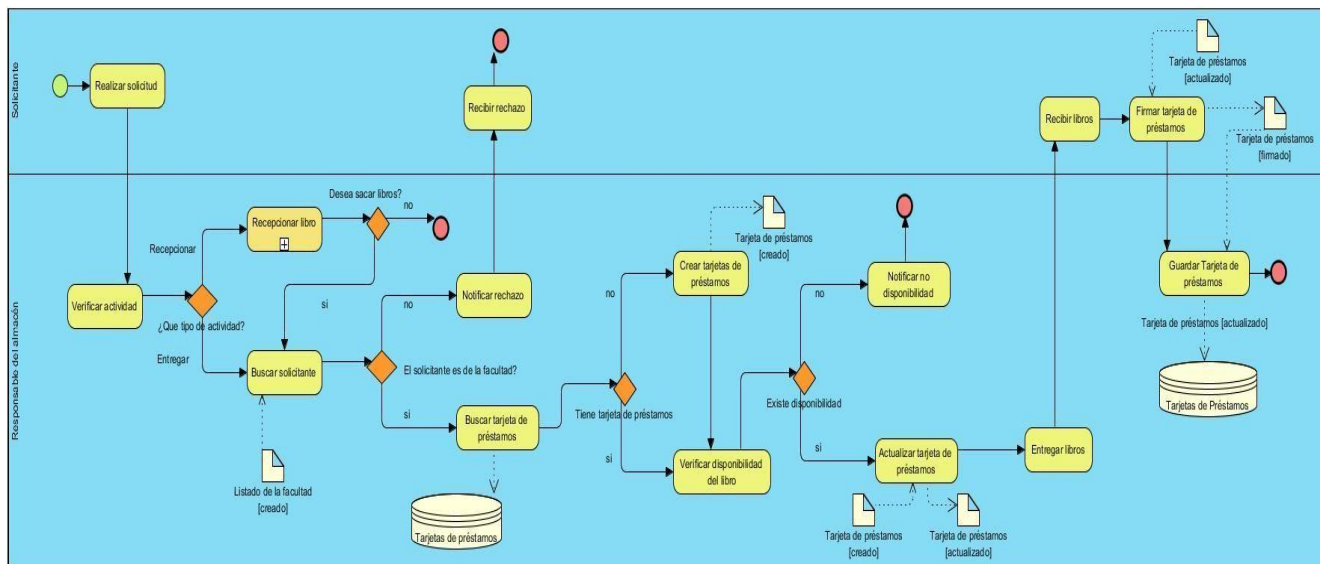


Ilustración 6: Descripción del subproceso: Entregar libros a estudiantes o trabajadores (Elaboración propia).

Hay que tener presente que también los estudiantes o trabajadores pueden hacer entrega de los libros que poseen. Por lo tanto, una misma persona puede solicitar libros y a su vez puede entregar. A continuación, se describe el proceso Recepcionar libros.

3. Recepcionar libros:

El responsable de almacén debe verificar si es un trabajador o un estudiante (ver Ilustración 7). En caso de ser un estudiante se realiza el subproceso Recepcionar libros entregados a los estudiantes y en caso contrario se realiza el subproceso Recepcionar libros entregados a los trabajadores.

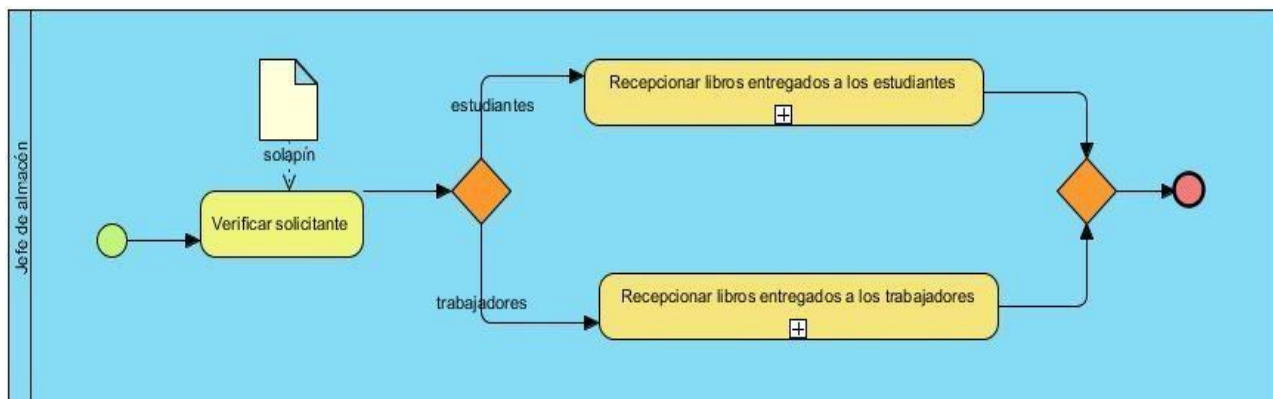


Ilustración 7: Descripción del proceso: Recepcionar libros (Elaboración propia).

• Recepcionar libros entregados a los estudiantes

En caso de que el estudiante desee entregar libros el responsable del almacén debe recepcionar los libros, actualizar la tarjeta de préstamo y el estudiante debe firmar el recibo (ver Ilustración 8). Si el estudiante se encuentra en proceso de baja y les faltan libros entonces el responsable del almacén debe llenar el modelo de débito en el que se incluyen los siguientes datos: nombre y apellidos, número de solapín, grupo, cantidad de libros que debe. Por cada libro que debe se deben registrar los siguientes datos: título del libro, importe en cup y el importe total a cobrar en cup.

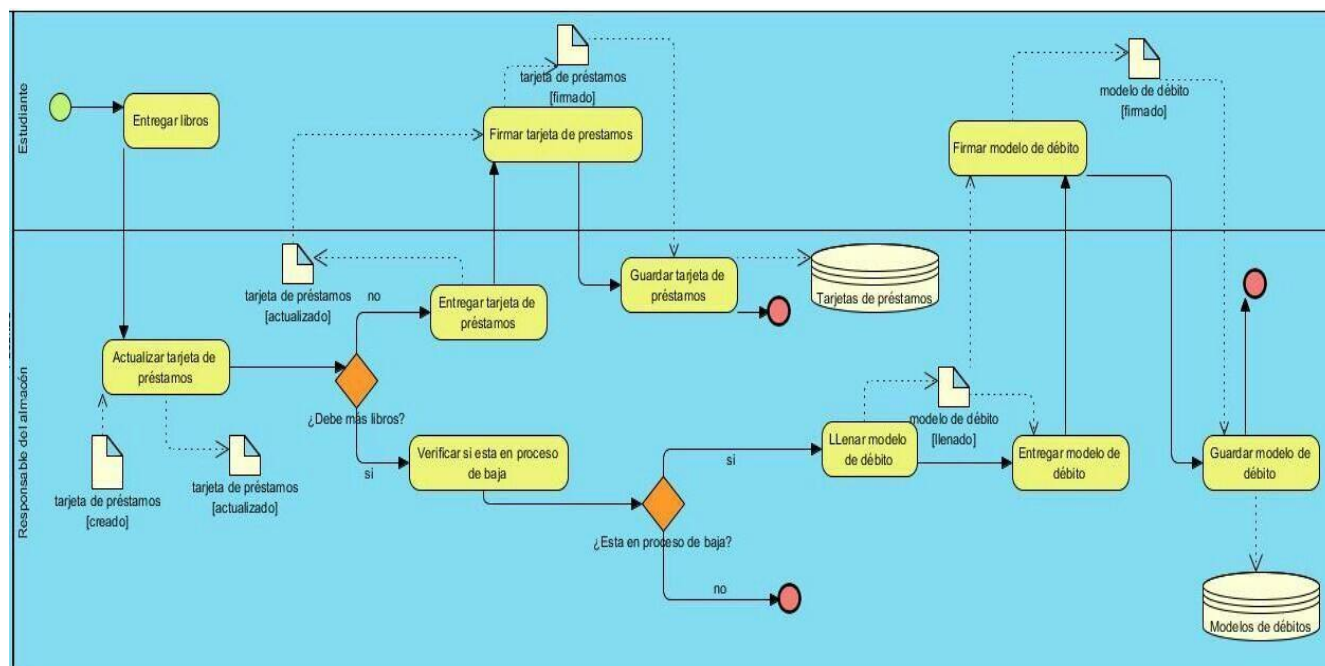


Ilustración 8: Descripción del subproceso: Recepcionar libros entregados a estudiantes (Elaboración propia).

• Recepcionar libros entregados a los trabajadores

En caso de que el trabajador desee entregar libros, el responsable del almacén debe recepcionar los libros y actualizar la tarjeta de préstamo, para posteriormente el trabajador firmar el recibo (ver Ilustración 9). En caso de que el trabajador se encuentre en proceso de baja, cuando el Vicedecano de Administración y Economía recibe la solicitud de baja, debe notificar al responsable de la aceptación de la baja si el trabajador no debe libros. Si debe libros se realiza el mismo flujo de actividades descrito anteriormente.

4. Controlar los libros en el almacén

13

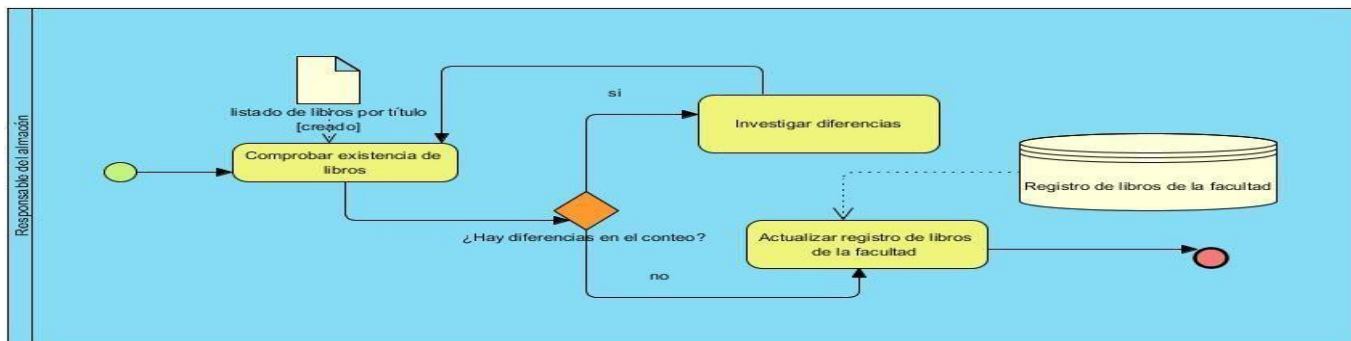


Ilustración 10: Descripción del subproceso: Controlar los libros en el almacén (Elaboración propia).

5. Control de recursos

El Vicedecano de Administración y Economía puede realizar controles a los recursos (de limpieza y materiales) en cualquier momento (ver Ilustración 11). La cantidad de recursos existentes debe coincidir con la diferencia de lo que se le entregó a la facultad (de acuerdo con los VS archivados) menos la cantidad de recursos entregados (de acuerdo a las actas de destino final existentes). En caso de existir irregularidades se comprueba con los responsables (Administrador del docente o Asistente de Control) las causas y se actualizan las actas de destino final de ser necesario. Posteriormente, se le notifica las correcciones realizadas a las altas de destino final o las soluciones encontradas ante las irregularidades al Vicedecano de Administración y Economía, quien es el encargado de volver a comprobar la existencia de los recursos materiales.

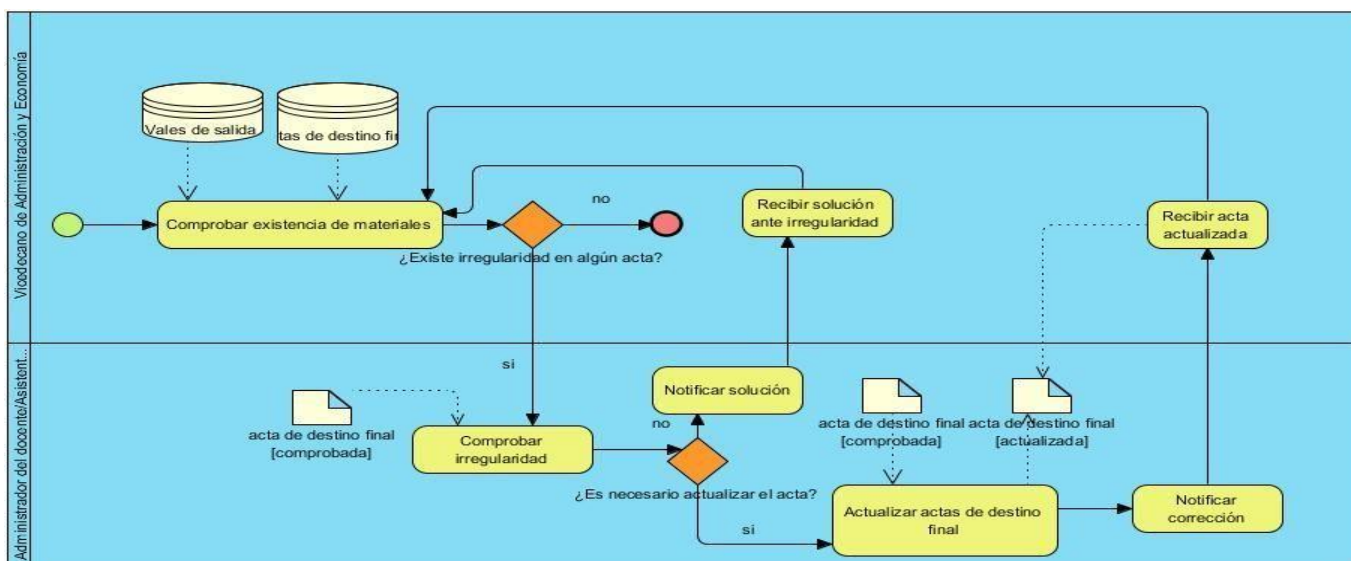


Ilustración 11: Descripción del subproceso: Control de recursos (Elaboración propia).

1.2.1 Análisis de los procesos Asignación y Control de recursos

En el análisis de los procesos se pudo determinar la existencia de varias deficiencias que afectan su ejecución en la facultad. A continuación, se hace mención de estas deficiencias:

- El histórico de la cantidad de recursos entregados a la facultad y asignados a los trabajadores y estudiantes se encuentran en archivos impresos por años, lo que dificulta la toma de decisiones y la revisión del destino final.
- Para el préstamo de libros se realizan tarjetas que son archivadas de forma física. Esto puede provocar que exista duplicidad de la información, extravío de tarjetas de préstamos y pérdida de libros.
- Las actividades de creación y actualización de las tarjetas de préstamos, modelos de débitos y el registro de libros de la facultad se realizan de forma manual provocando demora en el proceso.
- La gestión de cantidades se realiza de forma manual lo que dificulta la entrega de información, por ejemplo, saber de un título en específico la cantidad total (existencia en almacén + préstamos), conocer cuántos estudiantes de un grupo o un año no han entregado los libros o cuántos no han solicitado préstamos al almacén requiere un tiempo considerable al encargado de almacén.
- Actualmente, el vicedecanato de administración y economía no utiliza ninguna aplicación de negocio en el que se puedan gestionar estos procesos.

Ante las deficiencias encontradas es necesario realizar un estudio de herramientas informáticas en las que se evidencien la informatización de procesos que se asemejen a los descritos anteriormente. En una búsqueda bibliográfica se identificó que hay un tipo de sistema en particular que se utiliza en las empresas y son los sistemas de planificación de recursos empresariales (ERP, *Enterprise Resource Planning*). Estos sistemas se caracterizan por integrar y manejar procesos de diferentes áreas de una misma organización evidenciándose así la informatización de procesos relacionados con la logística, distribución, inventario, envíos, facturas y contabilidad (6).

Hay que enfatizar que la solución informática que se pretende desarrollar no es considerada un ERP ya que; solo estará enfocada a la mejora de algunos de los procesos que se realizan en el Vicedecanato de Administración y Economía de la facultad. Es decir, no permitirá la integración e informatización de otros procesos que se realizan en diferentes áreas de la

facultad, pero, por las características y funcionalidades que tienen algunos de los módulos que integran estos tipos de sistemas se tienen en cuenta en el estudio de herramientas similares. A continuación, se realiza un estudio de sistemas similares en los que se analiza su semejanza con los procesos de asignación y control de los recursos materiales en la Facultad 4.

1.3 Sistemas similares

El desarrollo de las TIC ha permitido la incorporación de herramientas que apoyan los procesos de gestión administrativos. A continuación, se describen algunos sistemas que presentan alguna similitud con los procesos de asignación y control de los recursos materiales. **Sistema Integrado de Gestión Administrativa (SIGA):** sistema informático que tiene gran importancia en instituciones públicas debido a que contribuye al ordenamiento y simplificación de los procesos de la gestión administrativa. Constituye uno de los programas más recomendables para la gestión de almacenes e inventarios. Ofrece múltiples funcionalidades: controla el inventario, las entradas y salidas manuales y programadas, reubicaciones, entre otras. Este sistema presenta grandes ventajas como: el ahorro de tiempo y costes que genera; programa adaptable a las necesidades de la organización (7).

SoftExpert Almacén: es un software enfocado en la gestión de existencias, depósitos e inventario. Posibilita a los usuarios el rápido y preciso acceso a las informaciones del historial de los eventos realizados en los locales de almacenaje de materiales, sus entradas y salidas. El sistema cuenta con herramientas de monitorización y control de niveles de existencias. Monitoriza las transacciones de inventario optimizando la gestión de materiales, insumos y residuos; reduce y controla los costos de procesos; mejora el control de recursos, coloca a disposición, listados con cantidad y materiales disponibles (8).

Koha: es un software libre y de código abierto de automatización de bibliotecas, desarrollado en arquitectura de cliente-servidor multiplataforma. Es un sistema basado en tecnologías web, flexible, que se adapta a requerimientos específicos de cada institución y además no exige recursos avanzados de tecnología, por lo que se puede instalar en equipos con características mínimas de hardware, lo que hace de éste un sistema económico de acceso libre (9). Es aplicable a los requerimientos de una biblioteca académica, escolar o pública. Facilita la gestión de préstamos, adquisiciones y permite obtener informes. Creado como una solución de software que permite la administración eficaz y eficiente de las funciones de: circulación, consulta, adquisición y descarte.

Biblio: programa destinado a la gestión y administración de los libros en centros educativos. Entre sus funcionalidades se encuentra la de gestión de préstamos, listados, altas de libros y más. Constituye una aplicación muy ligera que requiere menos espacio como la media de los programas de la categoría utilidades y herramientas. Posee listados con los libros más solicitados y los libros prestados (10).

SGA AHORA: sistema ERP que constituye uno de los programas de gestión de almacén más avanzados en la actualidad, donde toda actividad relacionada con el almacén del negocio puede centralizarse en este software, el cual registra los datos de cada movimiento. Permite la reducción de tiempo, menos personal y resultados de una operativa eficiente con bajo límite de error, convirtiéndolo en un programa automatizado bastante eficiente. Entre sus características se encuentra: la simplificación de procesos, el manejo de los datos de productos, etiquetado, permitidos y prohibidos. Realiza con eficiencia el control de almacenamiento y existencias, dando respuesta a necesidades importantes, además de una planificación de entregas óptima. Toda la información es accesible desde el punto de vista administrativo, evitándose la pérdida de tiempo. Permite la reducción de costes de almacenaje, mejora la información y la toma de decisiones, conocer el *stock* y registro de entradas y salidas, mejora la productividad del almacén y sus operarios (11).

Compass Suite: es un sistema de planificación de recursos empresariales basado en Oracle, diseñado para admitir, automatizar, administrar, controlar y supervisar los procesos y funciones empresariales dentro de una empresa mediante el uso de diferentes módulos integrados que representan a todos los departamentos de la organización (12).

SalesBabu ERP: permite a los usuarios monitorear, planificar, analizar y ejecutar. SalesBabu ERP tiene características como sincronizar niveles de inventario, mantener existencias, consumir menos tiempo, centralizar la administración, visibilidad del estado, optimizar las operaciones de almacén, ahorrar tiempo para conciliar datos, actualización de existencias en tiempo real, fácil informe de inventario a análisis, gestión integrada de pedidos, fácil acceso a información centralizada, reducir registros duplicados, entre otras (12).

1.3.1 Análisis del estudio de los sistemas similares

En el caso de los sistemas ERP que se describen: SGA AHORA, *Compass Suite* y *SalesBabu ERP*, no constituyen en primer lugar una solución para la problemática de investigación planteada, debido a que estos se caracterizan por integrar y manejar procesos de diferentes

áreas de una misma organización, y constituyen sistemas de pagos. A pesar de esto, presentan algunas funcionalidades relacionadas con la gestión de inventario y control de recursos, por lo que se tuvieron en cuenta para la definición de los requisitos funcionales. En cuanto al sistema Sistema Integrado de Gestión Administrativa (SIGA), es un sistema *Desktop* que solo puede utilizarse en Windows, y uno de los requisitos definidos por el cliente, es que debe ser un sistema web debido a que tipos de sistemas presentan grandes ventajas como: usan menos recursos que los programas instalados, son fáciles de utilizar, facilitan el trabajo colaborativo y a distancia debido a que puede ser usada por varios usuarios a la vez, provocan menos errores y los datos son más seguros.

A pesar de esto, hay que tener en cuenta que SIGA presenta funcionalidades relacionadas con el control de recursos en los almacenes, los cuales pueden ser definidos en la propuesta de solución. Por ejemplo, en este sistema se definen requisitos relacionados con la gestión de tipos de recursos: adicionar tipo de recurso, eliminar recurso, ver tipo de recurso, modificar datos de un tipo de recurso, listar tipos de recursos; los cuales deberían tenerse en cuenta para definir los diferentes tipos de recursos (insumos de limpiezas, libros y otros), que la Facultad le entrega al personal (estudiantes, trabajadores, personal de servicio).

En el caso del sistema Biblio de igual manera es un sistema *Desktop*, pero presenta funcionalidades que se encuentran relacionadas con el control de préstamos de libros. Estas funcionalidades son de gran relevancia para el negocio en cuestión y por consiguiente se van a tener en cuenta en la definición de los requisitos funcionales.

Por último, el sistema *Koha*, es un sistema integrado de gestión bibliotecaria multiplataforma, basado en tecnologías *Web* que puede ser utilizado por las bibliotecas para la ejecución de cada uno de sus procesos. Al igual que Biblio, no constituye una solución al problema planteado, pero presenta requisitos relacionados con el control de libros, que se van a tener en cuenta en el desarrollo de la propuesta de solución.

1.4 Metodologías de desarrollo de software, lenguajes y herramientas para el modelado

En esta sección se define la metodología de desarrollo de software a utilizar, y que constituirá la guía del proceso de desarrollo. Además, se definen los lenguajes y las herramientas necesarias para el modelado de la solución.

1.4.1 Metodología de desarrollo de software

Las metodologías de desarrollo de software constituyen un conjunto integrado de técnicas y métodos que permite abordar cada una de las actividades del ciclo de vida de un proyecto de desarrollo. Definen artefactos, roles y actividades, junto con prácticas y técnicas recomendadas (13). Las metodologías se dividen en tres grupos que engloban todas las variaciones de estas: las metodologías ágiles, las metodologías tradicionales y las metodologías híbridas. Las metodologías tradicionales centran su atención en llevar una documentación exhaustiva de todo el proyecto, la planificación y control del mismo, en especificaciones precisas de requisitos y modelado y en cumplir con un plan de trabajo. Como ejemplos de esta metodología tenemos: RUP (*Rational Unified Process*) y MSF (*Microsoft Solution Framework*).

Las metodologías ágiles basan su fundamento en la adaptabilidad de los procesos de desarrollo. Están orientadas a la interacción con el cliente y el desarrollo incremental del software, mostrando versiones funcionales del software al cliente en determinados períodos de tiempo, para que pueda sugerir cambios y dar una evaluación. Entre los ejemplos de esta metodología se encuentran: XP (*Extreme Programming*), Scrum y AUP (*Agile Unified Process*). Las metodologías híbridas combinan las mejores prácticas de las metodologías tradicionales y ágiles, incluyendo las ventajas de ambas. Se adaptan a la cultura y necesidades de las entidades, lo que genera un incremento de productividad. Presentan grandes beneficios como su simplicidad y fácil manejo, flexibilidad a cambios dentro de un alcance de presupuesto definido y una mejora continua exponencial (14).

Como ejemplo de esta metodología tenemos:

- **AUP-UCI:** constituye una variación del Proceso Unificado Ágil de Scott Ambler (AUP, *Agile Unified Process*) en unión con el modelo CMMI-DEV v 1.3. Fue desarrollada en la Universidad de las Ciencias Informáticas (UCI) con el objetivo de obtener una metodología que se adaptara mejor a los proyectos que se realizan en la institución. Consta de 3 fases: inicio, ejecución y cierre; y, además, propone 7 disciplinas: modelado, requisitos, análisis y diseño, implementación, pruebas internas, pruebas de liberación y pruebas de aceptación (15).

Teniendo en cuenta lo antes descrito se seleccionó la metodología híbrida debido a que posibilita la fácil adaptación a las necesidades del proyecto, la reducción de incertidumbre y una minimización de los riesgos a través de la interacción con los usuarios. Dentro de ella se escogió la AUP-UCI por su adaptación a los proyectos de la UCI y la posibilidad de realizar las mejores prácticas en el desarrollo del sistema, debido a su apoyo en el Modelo CMMI-DEV v 1.3. También, propone escenarios para modelar el sistema a partir del modelado del negocio y las formas de encapsular los requisitos, seleccionándose para este proyecto el escenario 3: proyectos que modelen el negocio con diagrama de procesos de negocio (DPN) solo pueden modelar el sistema con descripción de requisitos por procesos (DRP), el cual se aplica a proyectos que hayan evaluado el negocio que se quiere informatizar y se obtenga como resultado un negocio con procesos muy complejos e independientes de los ejecutores.

1.4.2 Lenguaje de modelado

Un lenguaje de modelado es un conjunto de elementos o notificaciones que están disponibles para el apoyar el modelado de una parte de una aplicación. Suelen estar ligados a metodologías de desarrollo. A continuación, se muestra algunos de los tipos de lenguajes de modelado existentes:

Lenguaje Unificado de Modelado (UML, *Unified Modeling Language*): lenguaje gráfico para la especificación, visualización, construcción y documentación de modelos orientados a objetos que representan sistemas intensivos en software (16). Está compuesto por tres componentes principales: elementos del modelo, relaciones entre elementos del modelo y diagramas UML (17).

Constituye un programa capaz de abstraer cualquier tipo de sistema y para lograrlo utiliza diferentes tipos de diagramas entre ellos (17): diagrama de clase, diagrama de componentes, diagrama de implementación, diagrama de objetos, diagrama de paquetes, diagrama de estructura de composición y diagrama de perfil.

Es independiente de cualquier lenguaje y cuenta con su propia metodología de desarrollo basada en componentes. Da soporte para la orientación a objetos, diseñar y aplicar marcos y patrones. Permite generar mayor extensibilidad e incorpora mejores prácticas de conformidad con los estándares de la industria (17).

Notación para la gestión de procesos de negocios (BPMN, *Business Process Modeling Notation*): es un lenguaje de diagramas de procesos. Describe en una imagen, las etapas de un proceso de negocio desde el principio hasta el fin. Se centra en los procesos funcionales, dejando el desarrollo de las interfaces a un lado. Las primitivas conceptuales básicas en los modelos BPMN son eventos, tareas y flujos. Los modeladores BPMN existentes permiten diseñar diagramas fácilmente (18).

A través del estudio realizado a estos lenguajes de modelado se llegó a la conclusión de la selección de ambos lenguajes para la realización del modelado del sistema. El lenguaje UML se utilizaría para la especificación, visualización y documentación de los modelos de sistema de software, ofreciendo un estándar para la descripción del modelo de sistema, con la inclusión de aspectos conceptuales y concretos como: la realización del modelo entidad-relación del diagrama de clases del diseño y el modelo de implementación. Además, de este lenguaje, se emplearía el BPMN para facilitar la comprensión del negocio a través de la definición de un único diagrama: el de procesos del negocio.

1.4.3 Herramienta de modelado

Para modelar los artefactos de la propuesta de solución se utilizan las potencialidades que brinda Visual Paradigm. Esta constituye una herramienta CASE¹ multiplataforma de modelado visual que soporta una amplia gestión de casos de usos y diseño de base de datos relacionales. Brinda grandes facilidades para la creación de los diferentes diagramas y cuenta con la posibilidad de realizar un control de versiones durante todo el ciclo de trabajo. Admite exportar e importar ficheros de proyectos realizados además de los artefactos que genera. También, facilita el desarrollo de ingeniería inversa y la generación automática de informes en formato PDF, Word o HTML (19).

¹ CASE: Computer Aided Software Engineering (Ingeniería de Software Asistida por Computadora). Referida al mantenimiento de proyectos software con la ayuda de diferentes herramientas automatizadas.

1.5 Lenguajes y Tecnologías de programación

1.5.1 Lenguajes de programación

Un lenguaje de programación es un conjunto de secuencias de instrucciones empleado en la creación de programas que son ejecutados a través de los ordenadores, lo que permite el control del comportamiento lógico del mismo (20). Constituyen una herramienta que se puede emplear en el desarrollo web, con el fin de crear sitios web que funcionen de forma más dinámica y para gestionar la lógica avanzada (21).

A continuación, se hace un estudio de tres lenguajes: *PHP*, *Python* y *JavaScript*, con el objetivo de obtener el que más se ajusta a los requerimientos del desarrollo del sistema. **PHP:** lenguaje interpretado con una sintaxis similar a la de C++ o Java. Se puede usar para crear cualquier programa, pero donde ha alcanzado mayor popularidad es en el desarrollo de páginas web. Su código se procesa en el servidor. Constituye un lenguaje libre, de alto nivel, disponible para muchos sistemas. Contiene múltiples extensiones: para conectar con base de datos, manejo de *sockets*, generar documentos PDF y páginas en Flash, entre otros (22). **Python:** lenguaje de programación orientada a objetos de alto nivel. Constituye uno de los lenguajes más rápidos y que requiere de muy pocas líneas de código. Ofrece soporte para módulos y paquetes, que permite modularidad en el sistema y la reutilización de códigos. Contiene estructuras de datos integradas, enlaces dinámicos y tipo, ideal para el desarrollo rápido de aplicaciones (23).

JavaScript: lenguaje de programación orientada a objetos e interpretado, es decir, que no requiere de compilación. Se encuentra basado en prototipos y multiparadigma. Es el lenguaje de programación web más utilizado en todo el mundo. Cuenta con una gran cantidad de *frameworks* que le permiten versatilidad en los resultados (24).

A continuación, se realiza una comparación de los tres lenguajes de programación antes mencionados, teniendo en cuenta las ventajas, desventajas y los *frameworks* que utilizan:

Tabla 1: Comparación de los lenguajes de programación Python, PHP y JavaScript [Tomado de: (25)]

Lenguajes de programación	<i>Python</i>	<i>PHP</i>	<i>JavaScript</i>
Ventajas	El intérprete de <i>Python</i> y la extensa	Su sintaxis es muy similar a la de otros	Lenguaje de <i>scripting</i> seguro y fiable.

	<p>biblioteca estándar están a libre disposición y es de código fuente abierto (26). Gran cantidad de funciones y bibliotecas como: <i>Numpy</i>, <i>Scipy</i>, <i>TensorFlow</i> y <i>QISKit</i> (26). Multiplataforma y portable.</p>	<p>lenguajes. Fácil de aprender. Capacidad de conexión con la mayoría de los gestores de base de datos: <i>PostgreSQL</i> y <i>Oracle</i>. No requiere definición de tipos de variables. Multiplataforma. Es rápido y bastante documentado.</p>	<p>Los <i>scripts</i> tienen capacidades limitadas, por seguridad. Su código se ejecuta del lado del cliente y del servidor.</p>
Desventajas	<p>Los lenguajes interpretados suelen ser relativamente lentos.</p>	<p>Al ser interpretado en el servidor, puede colapsar cuando el número de peticiones de descarga de páginas aumente.</p>	<p>Código visible para cualquier usuario. El código debe ser descargado completamente. Por lo general debe combinarse con otros lenguajes para poder diseñar un sitio web completo. Algunos buscadores no son capaces de acceder al contenido Javascript de la página lo que dificulta su posicionamiento.</p>
<i>Frameworks</i>	<p>Cuenta con pocos <i>frameworks</i> y se encuentran en constante mejoramiento: <i>Django</i> <i>Pyramid</i> <i>Pylons</i> <i>Flask</i> <i>Bottle</i></p>	<p>Utiliza <i>frameworks</i> con una estructura establecida y madura: <i>Zend</i> <i>Symfony</i> <i>Phalcon</i> <i>CodeIgniter</i> <i>Laravel</i></p>	<p><i>Angular</i> <i>Ember</i> <i>Backbone</i> <i>React</i></p>

Al concluir con el estudio de algunos de los lenguajes de programación se seleccionaron dos lenguajes: *JavaScript* y *Python*. *JavaScript* se utilizará para añadir las características interactivas al sistema, mientras que *Python* se escoge como el lenguaje principal debido a

que se encuentra preparado para realizar cualquier tipo de programa, como aplicaciones de *Windows*, servidores web y páginas web. Es compatible con *frameworks* como *Pyramid* y *Django*, y *micro-frameworks* como *Bottle* y *Flask*. Estos *frameworks* son flexibles, escalables, seguros, y contienen varios módulos y bibliotecas estándares útiles para la simplificación de tareas como la interacción con base de datos y la gestión de contenidos (27). Además, *Python* contiene varias bibliotecas como: *Matplotlib* para generar gráficos de calidad, *Seaborn* para la visualización de los datos y *NumPy* para la preparación de los datos y cálculo de atributos relevantes.

1.5.2 Frameworks de desarrollo

Un *framework* constituye un conjunto de conocimientos y técnicas que permite la utilización racional de los materiales y de los recursos naturales, mediante invenciones, construcciones u otras realizaciones que sean provechosas para el hombre. En el caso de los *frameworks* para aplicaciones web, constituyen una estructura definida que permite la reutilización de sus componentes, facilitando la creación de las aplicaciones web, posibilitando el ahorro de tiempo y mantenimiento (28).

En este caso como se escogió *Python* como lenguaje de programación principal, algunos de los *frameworks* que utiliza son: *Pyramid*, *Flask*, *Bottle* y *Django*. Teniendo en cuenta diferentes factores, se realiza una comparación a los *frameworks* antes mencionados (20) (29) (30):

Tabla 2: Comparación de los frameworks de Python

	<i>Django</i>	<i>Flask</i>	<i>Pyramid</i>	<i>Bottle</i>
Habilidades requeridas	<i>Python</i>	<i>Python</i>	<i>Python</i>	<i>Python</i>
Documentación	Buena documentación. Cuenta con una gran comunidad que se encuentra activa.	Buena documentación.	Documentación completa y actualizada.	Buena documentación.
Arquitectura	<i>Model-Template-View (MTV)</i> .	De acuerdo con los requerimientos del proyecto puede iniciar con el patrón MVC o utilizar otro.	Modelo-VistaControlador (MVC).	Modelo-VistaControlador.

ORM ² y conexiones a base de datos	Utiliza un ORM para asignar sus objetos a tablas de base de datos. Trabaja con PostgreSQL, MySQL, SQLite y Oracle.	No tiene ORM, pero si puede usar extensión. Integración con base de datos como Neo4j o base de datos noSQL como <i>Cloud Firestore</i> .	Libertad para el acceso a datos. Integración con base de datos SQL y noSQL.	Integración con bases de datos.
Herramientas y bibliotecas	Cuenta con su propio lenguaje de plantillas. Ofrece formularios <i>modelbased</i> . Variedad de bibliotecas integradas.	No contiene una base de datos especificada, por lo que puede usar MySQL o noSQL.	Proporciona solo herramientas básicas necesarias para casi todas las aplicaciones web: asignación de URL al código, seguridad, archivos <i>JavaScript</i> y CSS. Las herramientas adicionales brindan plantillas e integración de base de datos.	Incluye su propio motor de plantillas HTML simple. Presenta un sistema de enrutamiento que asigna URL s funciones, que contiene casi la misma sintaxis que <i>Flask</i> .
Aplicaciones	Aplicaciones web complejas que pueden ser grandes y escalables.	Aplicaciones web pequeñas o medianas.	Aplicaciones web básicas que pueden ser pequeñas y grandes.	Aplicaciones web pequeñas y simples.

Teniendo en cuenta el análisis realizado anteriormente se seleccionó como *framework* a utilizar para el desarrollo de la solución a *Django* debido a que proporciona un rápido desarrollo en la creación de aplicaciones web básicas y complejas, gracias a la creación de toda la estructura del *backend* de una aplicación web. También, cuenta con estructuras de seguridad de datos

² Mapeo objeto relacional (ORM, *Object-Relational Mapping*): permite convertir los datos del objeto en el formato correcto para poder guardar la información en la base de datos.

para salvaguardar la integridad de los *backend* y genera menos números de líneas de código en comparación con los otros *frameworks*.

Además, es un *framework* muy utilizado por los desarrolladores debido a que su interfaz puede ser adaptable según las necesidades y los requisitos que tenga el desarrollador. Per-

mite verificar que el marco de trabajo se mantenga funcionando a pesar de estar sometido a cambios, haciendo de este *framework* el más efectivo (28).

1.5.3 Entorno de desarrollo

El entorno de desarrollo integrado (IDE, *Integrated Development Environment*) son los programas o interfaces visuales de programación que se utilizan para interactuar con el compilador y depurador de un lenguaje de programación. A continuación, se realiza una comparación de algunos entornos de desarrollo integrado que han sido desarrollados para *Python*:

Tabla 3: Comparación de los IDE de Python [Tomado de: (31)]

IDE	<i>IDLE</i>	<i>PyCharm</i>	<i>VIM</i>	<i>Sublime Text</i>
Características	Editor de código fácil de usar, versátil. Interfaz de ventanas múltiples. Kit de herramientas de GUI ³ de <i>Tkinter</i> ⁴ .	Multiplataforma. Desarrollo remoto. Soporte de base de datos. Herramientas de desarrollo integradas. Modo de edición en tiempo real. Incorpora varias bibliotecas para ayudar a los desarrolladores a explotar más opciones disponibles.	Su <i>script</i> permite que casi todas las tareas de programación se realicen en <i>Python</i> . <i>Ctags</i> ⁵ , para ayudar a lograr una mejor navegación de etiquetas. Completar vocabulario básico. Compatible con <i>Windows</i> , <i>Linux</i> , <i>IOS</i> ,	Editor simple, versátil y conveniente. Es ampliamente utilizado y se puede utilizar en diferentes plataformas. Compatibilidad con <i>Windows</i> , <i>Linux</i> , <i>MacOs</i> . Navegación sencilla para un alto rendimiento. Admite diferentes paquetes para

³ GUI: Interfaz gráfica de usuario.

⁴ *Tkinter*: interfaz por defecto de Python para el toolkit de la GUI Tk (50).

⁵ *Ctags*: herramienta de programación que genera un archivo de etiquetas para guardar todas las definiciones dentro del código (51).

			<i>Android, MacOS.</i>	personalizar el editor.
Ventajas	Permite a los usuarios resaltar errores, codificarlos automáticamente e identificarlos con precisión. Su función de	Admite frameworks como <i>Pyramid, Flask</i> y <i>Django</i> . Proporciona la función de código inteligente, realizando co-	Con una interfaz de teclado, puede mejorar la eficiencia. Admite varios complementos como herramientas de con-	Puede manejar múltiples lenguajes de marcado. Permite que los usuarios elijan el proyecto en el que

	corrección incorporada puede mejorar el rendimiento. Permite que los usuarios busquen y reemplacen archivos en el editor.	recciones de errores más precisas y rápidas. Con la ayuda de las conexiones de <i>Docker</i> y <i>Vagrant</i> y el terminal ssh, permite el desarrollo de programas en hosts remotos.	trol de versiones y herramientas de gestión de archivos. Realiza diferentes funciones extendidas a través de archivos .vimrc.	quieren trabajar. Posee herramientas importantes que proporcionan una amplia indexación de clases y funciones. Proporciona una API potente y un ecosistema organizado
--	---	---	---	---

				para lograr un alto rendimiento.
Desventajas	No se admite la copia a la biblioteca. El <i>script</i> no puede exceder las 100 líneas de código.	Interfaz compleja. Alto costo de sus ventajas.	La interfaz es torpe y algunos desarrolladores la encuentran inconveniente de usar. Su editor modal requiere que los usuarios dediquen un tiempo a aprender.	Es muy avanzado para los novatos. Su complemento <i>git</i> no es particularmente poderoso.

En este trabajo se decidió usar como IDE *Pycharm* debido a que ofrece soporte de primer nivel para varios marcos de trabajo de desarrollo web como *Python*, *JavaScript*, HTML/CSS, entre otros. Ofrece una gran compatibilidad específica para cada *framework* de desarrollo web como *Django*, *Flask*, *Pyramid*. Proporciona una asistencia inteligente a la codificación.

Además, funciona en *Windows*, *macOs* o *Linux*.

1.6 Sistemas Gestores de Base de Datos

Un sistema de gestión de base de datos constituye un conjunto de programas a través de los cuales se puede acceder, administrar y gestionar la información almacenada en una base de datos, que no es más que un conjunto de información que pertenecen a un mismo contexto que se encuentran almacenadas de forma sistemática y que pueden ser utilizadas

posteriormente (32). A continuación, se realiza un análisis de las ventajas y desventajas de algunos sistemas de gestión de base de datos:

Tabla 4: Comparación de Sistemas Gestores de Base de Datos [Tomado de: (33)]

	<i>PostgreSQL</i>	<i>DB2</i>	<i>MySQL</i>	<i>Oracle</i>
Ventajas	Fácil de administrar. Multiplataforma. Escalable. Realiza copias de seguridad en línea.	Permite tablas replicadas. Soporta todo tipos de datos.	<i>Open Source</i> . Velocidad al realizar las transacciones. Usa la licencia GPL.	Multiplataforma Monitorea en vivo los procesos. Presenta varias opciones al realizar un <i>BackUp</i> . Fácil interacción con el sistema.
Desventajas	Es vulnerable sin protección adecuada.	Lentitud al crear y ejecutar transacciones. Demasiado consumo de memoria RAM. Costo elevado en licencias personales.	Varias utilidades no están documentadas.	Sintaxis más difícil de entender. Inhabilidad de implementar el procesamiento recursivo.

Analizadas estas ventajas y desventajas se escoge a PostgreSQL, debido a su facilidad de administración y aprendizaje. Presenta sintaxis intuitiva y la capacidad de realizar copias de seguridad en línea en comparación con otros sistemas de gestión de bases de datos. Además, es multiplataforma y escalable, lo cual es ideal para el desarrollo web.

1.7 Conclusiones del capítulo

Con el análisis del negocio se identificó que en la actualidad no existe ninguna aplicación de negocio que se emplee en la asignación y control de los recursos en la Facultad 4. Además, con el estudio de las herramientas similares se pudo apreciar que ninguna de las estudiadas se puede utilizar para dar solución al problema planteado, debido a que en su mayoría son propietarias y otras no cubren en su totalidad las necesidades del cliente.

Todo el proceso será controlado y orientado por la metodología de desarrollo de software *AUP-UCI* en su escenario 3. Como herramienta de modelado se utilizará *Visual Paradigm*, empleándose como lenguaje de modelado *UML* y *BPMN* para diseñar los diagramas necesarios para la comprensión del sistema a desarrollar. Se definió como lenguaje de programación *Python*,

donde se integrará el *framework Django*, en aras de proporcionar una mayor rapidez en el desarrollo del sistema y como *IDE* de desarrollo se escogió *Pycharm*.

CAPÍTULO II: DISEÑO DE LA SOLUCIÓN PROPUESTA AL PROBLEMA CIENTÍFICO

En este capítulo se realiza el análisis y diseño de la propuesta de solución al problema planteado. Como parte del análisis se realiza la especificación de los requisitos funcionales y no funcionales, así como la descripción de los mismos. Se define la arquitectura del software y los patrones de diseño que sustentarán todo el proceso de desarrollo del sistema. Además, se diseña el modelo de Base de Datos y se muestran las descripciones a través de tablas de las diferentes entidades que la conforman.

2.1 Disciplina de modelado del negocio

Teniendo en cuentas las deficiencias encontradas en el análisis del negocio realizado en el capítulo anterior, se proponen algunas posibles soluciones con el objetivo de que los procesos de asignación y control de los recursos materiales en la Facultad 4 se efectúen de forma eficiente. Para esto, se realizó un encuentro con la Vicedecana de Administración y Economía en el que se decidieron realizar modificaciones en los procesos teniendo en cuenta que se va a desarrollar un sistema de gestión basado en tecnologías web. A continuación, se muestra cómo quedaría el proceso Entregar libros a estudiantes o trabajadores teniendo en cuenta las mejoras obtenidas en conjunto con el cliente, el resto de los procesos se encuentran en los Anexos B: Modelo de los procesos To-Be.

Entregar libros a estudiantes o trabajadores

Para realizar una mejora en el proceso de entrega de libros, se tuvo en cuenta la entrega de libros por módulos. Es decir, se pueden entregar módulos de libros según el año académico, la asignatura o de forma independiente. Es por eso que en el sistema se van a estar desarrollando tres subprocesos relacionados con la entrega de libros, los cuales serían:

1. Entregar módulo por año académico

Seguidamente de que el estudiante realice la solicitud del módulo (ver Ilustración 12), el responsable del almacén debe buscar en el sistema el grupo al que pertenece el solicitante y comprobar si se encuentra en el listado de los pendientes por módulo. Posteriormente se verifica la disponibilidad de los módulos y si hay disponibilidad, se comprueba de que el estudiante tenga asociado una tarjeta de préstamos. En caso de tener una se actualiza o en caso contrario, se le crea una. Luego se le hace entrega del módulo y el estudiante debe fir-

mar un listado de entrega, que debe archivar de forma física para tener constancia de que recibió el módulo. Para realizar la entrega de módulo por asignatura, el proceso es el mismo para trabajadores y estudiantes.

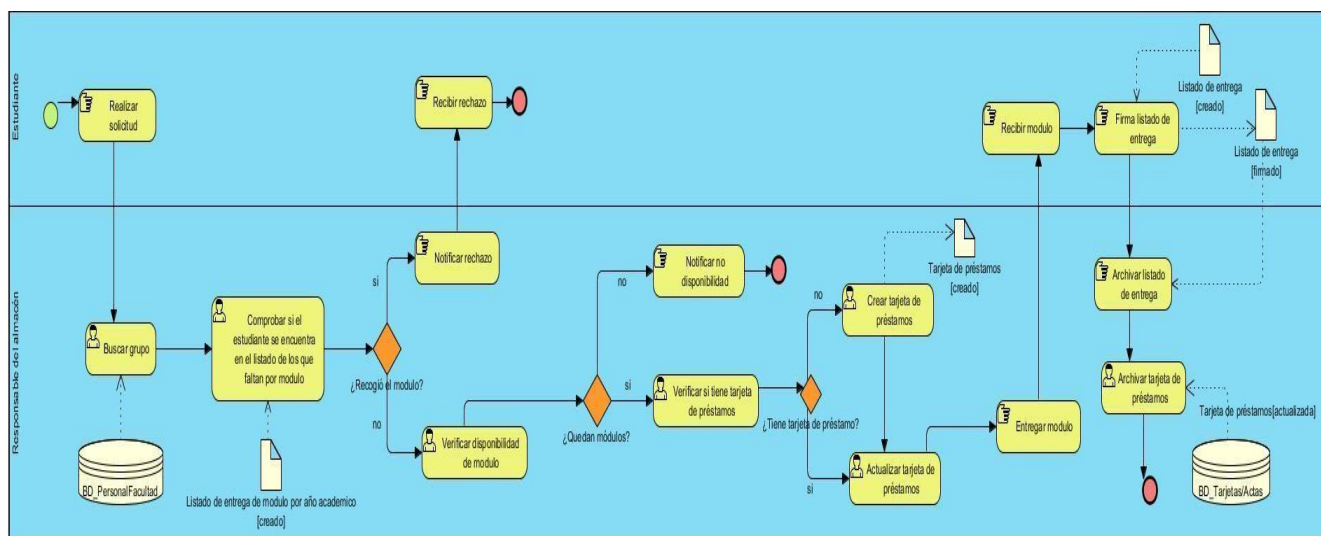


Ilustración 12. Modelo To-Be del proceso Entregar módulo por año académico a estudiantes (Elaboración propia).

2. Entregar libros

El responsable del almacén verifica la existencia del libro en el sistema (ver Ilustración 13). Si no existe, se le notifica al solicitante. En caso contrario, se verifica la disponibilidad del libro y si la respuesta es afirmativa entonces, el responsable debe pedir el solapín del solicitante. Si es estudiante, entonces el responsable debe buscar en el sistema el grupo al que pertenece y al estudiante dentro del listado de estudiantes del grupo. Posteriormente se comprueba de que tenga asociado una tarjeta de préstamos. En caso de no tenerla, se le crea una. Luego se actualiza la tarjeta, se le hace entrega del libro al solicitante, se imprime un listado de entrega que debe firmar el estudiante para posteriormente este ser archivado de forma física para tener constancia de que se le entregó el libro.

En caso de que el solicitante sea un trabajador se procede de la misma manera, pero en lugar del grupo, se comprueba el área de trabajo.

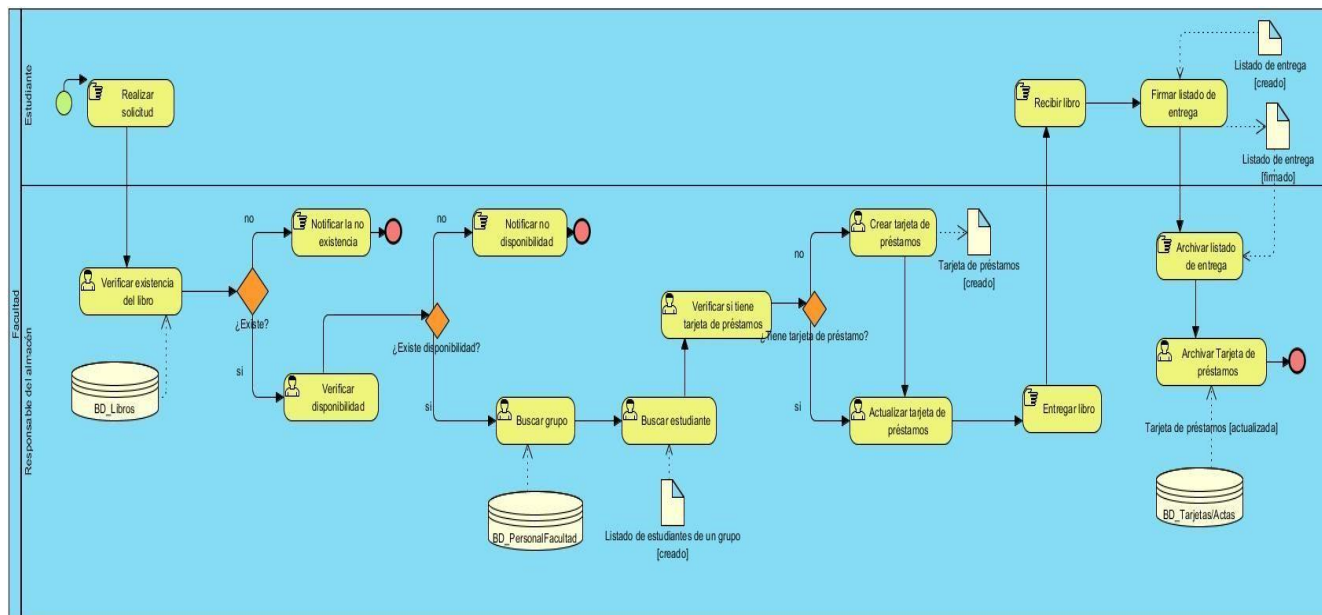


Ilustración 13: Modelo To-Be del proceso *Entregar libro a estudiantes* (Elaboración propia).

2.2 Propuesta de solución

Para dar solución a la problemática planteada en la presente investigación, se decide desarrollar un sistema para la asignación y control de los recursos materiales en la facultad. Este se concibió como un sistema basado en tecnologías web que cuenta con diferentes opciones como son crear tarjeta de préstamos, listar estudiantes de un grupo, visualizar los recursos materiales, crear módulos de materiales por año académico o por asignatura, entre otros que faciliten la interacción con la herramienta.

2.3 Disciplina de requisitos

La ingeniería de requerimientos constituye el proceso de descubrir, analizar y documentar los servicios y restricciones (34).

La técnica utilizada para la obtención de los requisitos fue la técnica de prospección a través de la entrevista, en este caso realizada a la Vicedecana de Administración y Economía de la Facultad 4. Esta técnica permite obtener y documentar de forma detallada la información sobre los requisitos. Además, direcciona al cliente hacia aspectos específicos del requerimiento a levantar (35).

Requisitos del sistema

Los requisitos son descripciones de lo que el sistema debe hacer: el servicio que ofrece y las restricciones en su operación. Reflejan las necesidades del cliente por un sistema que atienda

un propósito, como sería controlar un dispositivo o buscar información (34). Pueden ser clasificados en requisitos funcionales o requisitos no funcionales.

2.3.1 Requisitos Funcionales

Los requisitos funcionales (RF) son enunciaciones de los servicios que el sistema debe proveer, como debe reaccionar a entradas y su comportamiento ante situaciones particulares (35). A continuación, en la Tabla 5, se presentan los requisitos funcionales del sistema.

Tabla 5: Requisitos Funcionales.

#	Requisitos	Descripción
RF1	Adicionar grupo docente	El sistema debe permitir incluir un grupo docente. Los datos que se deben especificar son: Número del grupo Cantidad de estudiantes
RF2	Modificar grupo docente	El sistema debe permitir modificar los datos de un grupo docente.
RF3	Ver datos de un grupo docente	El sistema debe permitir ver los datos de un grupo docente.
RF4	Eliminar grupo docente	El sistema debe permitir eliminar un grupo docente.
RF5	Listar grupos docentes	El sistema debe permitir listar los grupos docentes existentes.
RF6	Adicionar estudiante	El sistema debe permitir adicionar un estudiante. Los datos del estudiante que se deben especificar son: Nombre Primer Apellido Segundo Apellido Grupo Docente No. Solapín CI
RF7	Modificar datos del estudiante	El sistema debe permitir modificar los datos del estudiante. Los datos del estudiante son: Nombre Primer Apellido Segundo Apellido Grupo Docente No. Solapín CI
RF8	Ver datos del estudiante	El sistema debe permitir ver los datos del estudiante.
RF9	Eliminar estudiante	El sistema debe permitir eliminar un estudiante.
RF10	Listar estudiantes de un grupo	El sistema debe permitir listar los estudiantes de un grupo.

RF11	Listar estudiantes por año	El sistema debe permitir listar los estudiantes por año.
RF12	Adicionar trabajador	El sistema debe permitir incluir un trabajador. Los datos para incluir de un trabajador son: Nombre Primer Apellido Segundo Apellido No. solapín Área Cargo CI
RF13	Modificar datos del trabajador	El sistema debe permitir modificar los datos de un trabajador. Los datos para modificar son: Nombre Primer Apellido Segundo Apellido No. solapín Área Cargo CI
RF14	Ver datos de un trabajador	El sistema debe permitir ver los datos de un trabajador.
RF15	Eliminar un trabajador	El sistema debe permitir eliminar un trabajador.
RF16	Incluir área	El sistema debe permitir incluir un área de trabajo. Los datos del área son: Nombre Cantidad de trabajadores Nombre del directivo del área
RF17	Ver área	El sistema debe permitir ver los datos de un área de trabajo.
RF18	Eliminar área	El sistema debe permitir eliminar un área de trabajo.
RF19	Listar áreas	El sistema debe permitir listar las áreas de trabajo existentes.
RF20	Listar los trabajadores de un área	El sistema debe permitir listar los trabajadores de un área de trabajo.
RF21	Adicionar tipo de material	El sistema debe permitir adicionar un tipo de material. Los datos que se deben especificar son: Nombre del tipo de material
RF22	Modificar tipo de material	El sistema debe permitir modificar los datos del tipo de material. Los datos son: Nombre del tipo de material

RF23	Eliminar tipo de material	El sistema debe permitir eliminar un tipo de material.
RF24	Ver tipo de material	El sistema debe permitir ver los datos de un tipo de material.
RF25	Listar tipo de material	El sistema debe permitir listar los tipos de materiales.
RF26	Adicionar recurso material	El sistema debe permitir incluir un recurso material. Los datos que se deben especificar son: Nombre del recurso material Tipo de material Cantidad existente Código
RF27	Modificar datos del recurso material	El sistema debe permitir modificar los datos del recurso material. Los datos del material son: Nombre del recurso material Tipo de material Cantidad existente Código
RF28	Eliminar recurso material	El sistema debe permitir eliminar un recurso material.
RF29	Ver datos de un recurso material	El sistema debe permitir ver los datos de un recurso material.
RF30	Listar recursos materiales	El sistema debe permitir listar los recursos materiales.
RF31	Crear módulo de materiales	El sistema debe permitir crear módulos de recursos materiales. El módulo estará conformado por varios materiales y se debe definir si será asignado a trabajadores o a estudiantes.
RF32	Eliminar módulo de materiales	El sistema debe permitir eliminar un módulo de materiales.
RF33	Modificar datos del módulo	El sistema debe permitir modificar los datos del módulo. Es decir, deberá permitir cambiar o eliminar recursos de un módulo, sus cantidades y la asignación de este.
RF34	Asignar módulo de materiales a estudiantes y trabajadores	El sistema debe permitir asignar módulo de materiales a los estudiantes y trabajadores. Cuando se realice una asignación se debe guardar la fecha en la que se realizó la misma.
RF35	Ver módulo de materiales	El sistema debe permitir ver los datos de un módulo de materiales.
RF36	Asignar materiales a estudiantes y a trabajadores	El sistema debe permitir asignarles materiales a los estudiantes y trabajadores. Cuando se realice una asignación se debe guardar la fecha en la que se realizó la misma.

RF37	Listar materiales por estudiantes	El sistema debe permitir listar los materiales entregados a los estudiantes.
RF38	Listar materiales por trabajador	El sistema debe permitir listar los materiales entregados a los trabajadores.
RF39	Listar estudiantes de un grupo docente que faltan por recoger el módulo	El sistema debe permitir listar los estudiantes que no han recogido el módulo de recursos materiales.
RF40	Listar trabajadores de un área que faltan por recoger módulo	El sistema debe permitir listar los trabajadores que no han recogido el módulo de materiales en un área de trabajo.
RF41	Generar acta de destino final	El sistema debe permitir generar un acta de asignación de materiales en cualquier momento. Se debe imprimir el listado del personal que ha recogido los recursos materiales.
RF42	Imprimir acta de destino final	El sistema debe permitir imprimir el acta de asignación de materiales.
RF43	Insertar usuario	El sistema debe permitir registrar un usuario. Los datos que debe registrar son: Usuario Contraseña
RF44	Modificar usuario	El sistema debe permitir modificar la información de un usuario. Los datos son: Usuario Contraseña
RF45	Eliminar usuario	El sistema debe permitir eliminar un usuario.
RF46	Listar usuario	El sistema debe permitir listar los usuarios.
RF47	Autenticar usuario	El sistema debe permitir autenticar al usuario.
RF48	Asignar rol	El sistema debe permitir asignar roles a usuarios en específico: Encargado de Almacén Observador (Profesor Principal, Vicedecano de Formación, Decano) Vicedecano de Administración y Economía

RF49	Modificar rol	El sistema debe permitir modificar los roles a usuarios en específico: Encargado de Almacén Observador (Profesor Principal, Vicedecano de Formación, Decano) Vicedecano de Administración y Economía
RF50	Eliminar rol	El sistema debe permitir eliminar el rol a un usuario en específico.
RF51	Adicionar libro	El sistema debe permitir adicionar un libro al material de estudio. Los datos que debe registrar son: Código Titulo Volumen Tomo Autor Cantidad Precio
RF52	Modificar libro	El sistema debe permitir modificar los datos de un libro, teniendo en cuenta: Código Titulo Volumen Tomo Autor Cantidad Precio
RF53	Eliminar libro	El sistema debe permitir eliminar un libro.
RF54	Listar libro	El sistema debe permitir consultar el catálogo de los libros existentes en el almacén, mostrando los siguientes datos: Titulo Autor Tomo Disponibilidad
RF55	Adicionar asignatura	El sistema debe permitir adicionar una asignatura. Los datos que debe ingresar son. Nombre Año Académico

RF56	Modificar asignatura	El sistema debe permitir modificar una asignatura, especificando los siguientes datos: Nombre Año Académico
RF57	Eliminar asignatura	El sistema debe permitir eliminar una asignatura.
RF58	Listar asignatura	El sistema debe permitir listar las asignaturas, mostrando los siguientes datos: Nombre

		Año Académico
RF59	Crear módulo de libros por asignatura	El sistema debe permitir adicionar un módulo de libros por asignatura, especificando los siguientes datos: Nombre del módulo de libros por asignatura Libros
RF60	Modificar módulo de libros por asignatura	El sistema debe permitir modificar un módulo de libros por asignatura, teniendo en cuenta los siguientes datos: Nombre del módulo de libros por asignatura Libros
RF61	Eliminar módulo de libros por asignatura	El sistema debe permitir eliminar un módulo de libros por asignatura.
RF62	Visualizar módulo de libros por asignatura	El sistema debe permitir visualizar los módulos de libros por asignaturas, especificando los siguientes datos: Nombre del módulo de libros por asignatura Libros
RF63	Crear módulo de libros por año académico	El sistema debe permitir adicionar un módulo de libros por año académico, teniendo en cuenta: Nombre del módulo de libros por año académico Libros Módulo de libros por asignatura
RF64	Modificar módulo de libros por año académico	El sistema debe permitir modificar un módulo de libros por año académico, especificando los siguientes datos: Nombre del módulo de libros por año académico Libros Módulo de libros por asignatura
RF65	Eliminar módulo de libros por año académico	El sistema debe permitir eliminar un módulo de libros por año académico.

RF66	Visualizar módulo de libros por año académico	El sistema debe permitir visualizar los módulos de libros por año académico, especificando los siguientes datos: Nombre del módulo de libros por año académico Libros Módulos de libros por asignatura
RF67	Crear tarjeta de préstamos	El sistema debe permitir crear una tarjeta de préstamos de libros a un solicitante. Cuando se realice el préstamo de un libro se debe disminuir la existencia de este en el almacén. Se debe especificar los siguientes datos:

		Nombre Primer Apellido Segundo Apellido Grupo o área de trabajo No. solapín Fecha del préstamo Fecha de devolución Libros Módulo de libros
RF68	Actualizar tarjeta de préstamos	El sistema debe permitir actualizar una tarjeta de préstamos. Los datos a actualizar son: Libros Fecha del préstamo Fecha de devolución Módulo de libros
RF69	Eliminar tarjeta de préstamos	El sistema debe permitir eliminar una tarjeta de préstamos.
RF70	Listar tarjetas de préstamos	El sistema debe permitir listar las tarjetas de préstamos por grupo o área de trabajo.
RF71	Generar modelo de débito	El sistema debe permitir obtener un modelo de débito cuando el usuario no pueda devolver un libro. El modelo debe contener: Nombre Primer Apellido Segundo Apellido Grupo o área de trabajo No. solapín Título del libro Importe
RF72	Listar los libros entregados a cada usuario	El sistema debe permitir mostrar el histórico de los libros solicitados por cada usuario, con fecha de entrega y devolución.

RF73	Generar reporte de préstamos	El sistema debe permitir generar un reporte de préstamos por año académico, grupo docente, área de trabajo o general.
RF74	Generar inventario de la base material de estudio	El sistema debe permitir generar un inventario de la base material de estudio. En este inventario de debe reflejar por cada título la cantidad que está en existencia en el almacén y la cantidad que se encuentra en poder de los estudiantes o trabajadores.
RF75	Generar modelo de no tenencia individual	El sistema debe permitir generar un modelo de no tenencia individual por cada persona que no contenga préstamos activos. Los datos que debe contener son: Nombre Primer Apellido Segundo Apellido Grupo o área de trabajo No. solapín

2.3.2 Descripción de requisitos

Teniendo en cuenta la metodología de desarrollo AUP-UCI, se va a realizar la descripción de los requisitos específicamente, la descripción de los requisitos por procesos (DRP). A continuación, se muestra la DRP del proceso: Crear tarjeta de préstamos

Tabla 6. DRP.1 Crear tarjeta de préstamos

Objetivo	Permitir crear una tarjeta de préstamos en el sistema.
Pre condiciones	<p>Estar autenticado en el sistema con el rol de responsable del almacén.</p> <p>Tener en cuenta los siguientes datos:</p> <ul style="list-style-type: none"> • *Nombre • *Primer Apellido • *Segundo Apellido • *Grupo o área de trabajo • *Solapín • *Fecha del préstamo • Fecha de devolución • *Libros • Módulo de libros

Comportamientos válidos y no válidos	<p>Los campos obligatorios se encuentran referenciados con un *.</p> <p>Nombre: campo de texto que admite caracteres alfabéticos y tiene un máximo de hasta 50 caracteres.</p> <p>Primer Apellido: campo de texto que admite caracteres alfabéticos y tiene un máximo de hasta 50 caracteres.</p> <p>Segundo Apellido: campo de texto que admite caracteres alfabéticos y tiene un máximo de hasta 50 caracteres.</p>
	<p>Grupo o área de trabajo: campo de selección.</p> <p>Solapín: campo de texto que admite caracteres alfanuméricos, con un máximo de 10 caracteres.</p> <p>Fecha de préstamo: campo que admite solo fecha.</p> <p>Fecha de devolución: campo que admite solo fecha.</p> <p>Libros: campo de selección en el que se pueden seleccionar uno o más libros.</p> <p>Módulos de libros: campo de selección.</p>
Flujo de eventos	
Flujo básico	
1. El usuario con el rol de responsable del almacén selecciona la opción Crear tarjeta de préstamos.	
2. El usuario introduce y/o selecciona los datos necesarios.	
3. El usuario selecciona la opción Guardar.	
4. Se crea una nueva tarjeta de préstamos y el sistema muestra el siguiente mensaje de confirmación: “La tarjeta se creó de forma correcta”.	
5. Se regresa a la página principal.	
Flujos alternos: “Selecciona la opción Cancelar”	
2.1 El usuario selecciona la opción Cancelar, mostrando el siguiente mensaje de confirmación: “Desea cancelar la operación”. Si el usuario selecciona la opción “Si” se regresará a la pantalla principal.	
Flujos alternos: “Datos incorrectos”	
4.1 Si el usuario selecciona la opción Guardar, pero existen datos que no corresponden con los especificados en los comportamientos válidos, se mostrará	

el siguiente mensaje de error: “Existen datos incorrectos, por favor rectifique” y se debe retornar al paso 2.

Flujos alternos: “Datos incompletos”

4.1 Si el usuario selecciona la opción Guardar, pero existen datos incompletos, se mostrará el siguiente mensaje de error: “Existen datos incompletos, por favor rectifique” y se debe regresar al paso 2.

Prototipo de Interfaz:

Nombre Primer Apellido Segundo Apellido CI

Solapín Grupo

Libros	Fecha del préstamo	Fecha de devolución
<input type="checkbox"/> Programación I	10/11/21 03:11 PM	10/11/21 03:12 PM
<input type="checkbox"/> Programación II	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Cálculo I	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Gestión de Software	10/11/21 03:12 PM	10/11/21 03:12 PM

Módulo por Asignatura

Módulo por Asignatura	Fecha del préstamo	Fecha de devolución
<input type="checkbox"/> Programación I	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Programación I	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Programación I	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Programación I	10/11/21 03:12 PM	10/11/21 03:12 PM

Módulo por Año Académico

Módulo por Año Académico	Fecha del préstamo	Fecha de devolución
<input type="checkbox"/> IA	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Gestión de software	10/11/21 03:12 PM	10/11/21 03:12 PM

Ilustración 14: Interfaz de DRP 1. Crear Tarjeta de Préstamos de estudiante (Elaboración propia).

Nombre Primer Apellido Segundo Apellido CI

Solapín Área de Trabajo

Libros	Fecha del préstamo	Fecha de devolución
<input type="checkbox"/> Programación I	10/11/21 03:11 PM	10/11/21 03:12 PM
<input type="checkbox"/> Programación II	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Cálculo I	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Gestión de Software	10/11/21 03:12 PM	10/11/21 03:12 PM

Módulo por Asignatura

	Fecha del préstamo	Fecha de devolución
<input type="checkbox"/> Programación I	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Programación I	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Programación I	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Programación I	10/11/21 03:12 PM	10/11/21 03:12 PM

Módulo por Año Académico

	Fecha del préstamo	Fecha de devolución
<input type="checkbox"/> IA	10/11/21 03:12 PM	10/11/21 03:12 PM
<input type="checkbox"/> Gestión de software	10/11/21 03:12 PM	10/11/21 03:12 PM

Ilustración 15: Interfaz de DRP 1. Crear Tarjeta de Préstamos de trabajador (Elaboración propia).

2.3.3 Requisitos no funcionales.

Los requisitos no funcionales (RNF) constituyen las propiedades que debe cumplir el sistema. Son restricciones sobre servicios y funcionalidades ofrecidos por el sistema (35).

RNF 1. Requisitos de software:

RNF 1.1. El sistema debe funcionar en cualquier sistema operativo: Windows, Linux.

RNF 2. Requisitos de restricciones de diseño e implementación:

RNF 2.1. Se utilizará como gestor de base de datos PostgreSQL 9.4 para el diseño del sistema.

RNF 2.2. El sistema se desarrollará en el lenguaje de Python 3.7.

RNF 2.3. Los artefactos ingenieriles se modelarán con la herramienta Visual Paradigm 8.0.

RNF 2.4. Para el desarrollo del sistema se utilizará el IDE de desarrollo Pycharm 2021.2.1.

RNF 3. Requisitos de seguridad:

RNF 3.1. El acceso a la información debe estar restringido según el rol que la utiliza.

RNF 3.2. Para enviar la información contenida en los formularios se utilizará el método POST debido a que realiza la modificación de los datos de forma segura del lado del servidor.

RNF 4. Requisitos de usabilidad:

RNF 4.1. El sistema debe revisar todos los campos de datos para comprobar que no se introduzcan datos erróneos y en caso de encontrar algún error, mostrar un mensaje que indique el error.

RNF 4.2. El sistema mostrará un mensaje con el resultado de la acción que se realizó, ya sea eliminar o modificar.

RNF 4.3. El sistema debe comprobar que todos los campos obligatorios del formulario se encuentren llenos.

2.4 Arquitectura del software

La arquitectura de software constituye la rama del desarrollo de software encargada de definir la interacción entre los datos, la lógica y la vista, y el papel que desempeña cada una de estos elementos en el software resultante (36).

Como la propuesta de solución se desarrollará con el *framework* Django se utiliza por defecto la arquitectura *Model Template View* (MTV), que no es más que una variación de la arquitectura Modelo Vista Controlador (MVC) (37):

- **Model** (Modelo): capa encargada de todo el acceso a la base de datos. Contiene todo lo relacionada a la línea de conexiones, referencias de datos y validaciones.
- **Template** (Plantilla): capa encargada de la presentación del aplicativo o página web.
- **View** (Vista): capa encargada de la parte lógica del negocio. Permite el acceso a la capa *Model* y es la encargada de realizar la conexión entre la capa *Model* y la *Template*.

Para una mejor comprensión de esta arquitectura del *framework Django* se realiza a continuación, los diagramas de clases del diseño.

2.4.1 Diagrama de clases del diseño

Los diagramas de clases se emplean para modelar un conjunto de elementos que representan la visión estática de un sistema. Soporta en general a los requisitos funcionales, pero en particular, a los servicios que el sistema debería proporcionar a sus usuarios finales. Está compuesto por: clases, interfaces y las relaciones entre estos, que pueden ser de dependencia, asociación o generalización (38).

A continuación, en la Ilustración 16, se muestran el diagrama de clases del diseño correspondiente al Gestionar libros, en el que se evidencia la arquitectura explicada con anterioridad. Otros diagramas de clases del diseño se encuentran en el Anexo C:

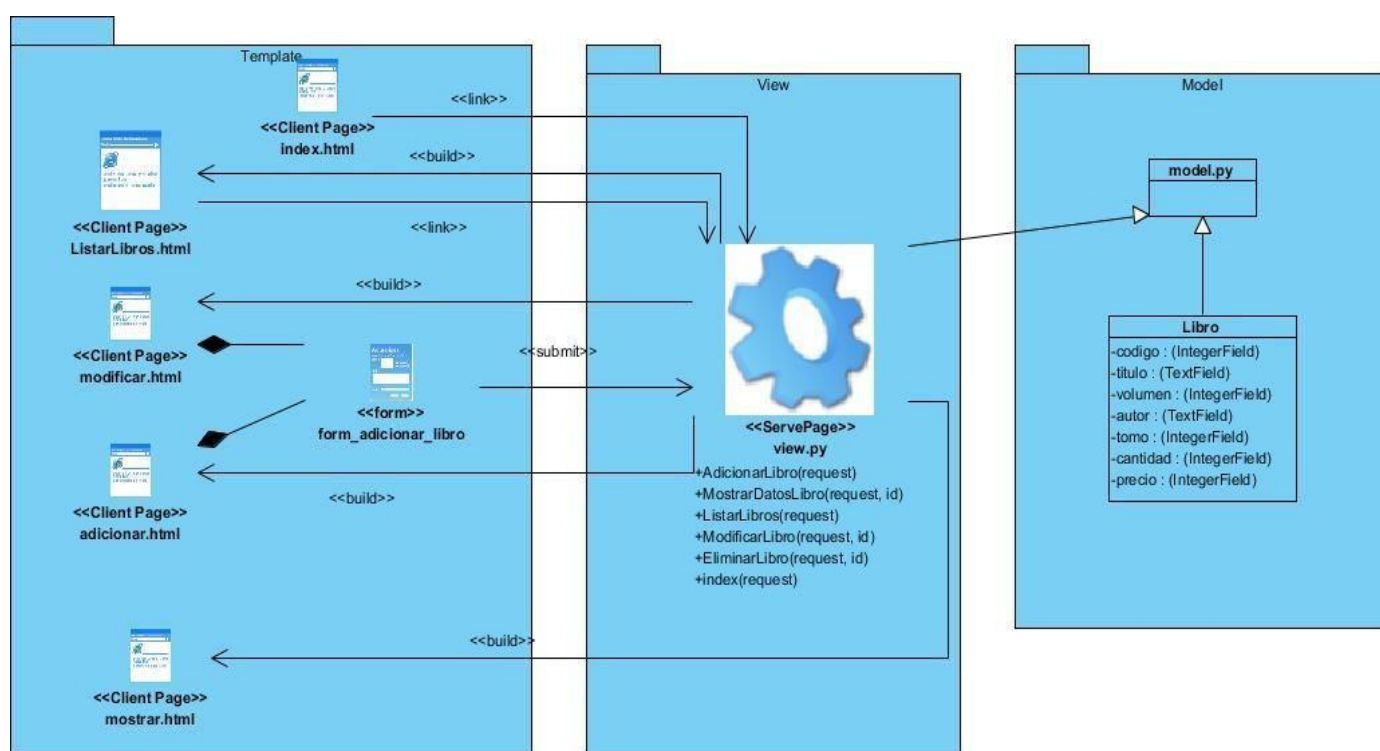


Ilustración 16: Diagrama de Clases del diseño Gestionar Libros (Elaboración propia).

En el diagrama de clases anterior se evidencia la utilización del patrón arquitectónico MTV el cual se encuentra estructurado en tres paquetes fundamentales (Template, View y Model). En el paquete Template están agrupadas todas las clases que se consideran plantillas o páginas HTML, que representan las vistas de la aplicación y sus formularios. En el paquete View se encuentra una sola clase en donde están contenidas todas las funciones que fueron implementadas en el lenguaje de programación Python que implementan la lógica del negocio.

Por último, en el paquete Model se identifican, en forma de clases, las entidades utilizadas para el desarrollo de la solución y que serán mapeadas con la base de datos.

Para un mejor entendimiento de la arquitectura a continuación se explica cómo funciona el requisito funcional Adicionar libro:

El Template ListarLibros.html brinda la opción de Adicionar un libro que, al acceder a ella, realiza un enlace a otro Template en este caso al Adicionar.html. Una vez que se selecciona esa opción el View recibe la petición y busca la función establecida para llevar a cabo la lógica de este requisito. En este caso la función realizada debe permitir agregar un nuevo libro en la base de datos del sistema. Una vez ingresado los datos en el formulario, la validación de los mismos está en correspondencia con las funciones definidas en las clases models de Python, que en este caso específico sería la clase Libro. De ser correctos entonces, se crea una nueva instancia de la clase Libro que se almacenará en la base de datos del sistema.

2.5 Patrones de diseño

Los patrones de diseño constituyen la base para la búsqueda de soluciones a problemas existentes que son muy comunes en el desarrollo de software y en otros escenarios referentes al diseño de interacción o interfaces, es decir, representa una solución a un problema de diseño (39).

2.5.1 Patrones GRASP

Los Patrones Generales de Software para Asignar Responsabilidades (GRASP, *General Responsibility Assignment Software Patterns*) son patrones que señalan la manera de asignar responsabilidades a los objetos del software a través de patrones (40).

Seguidamente se muestra la explicación de los patrones utilizados en la solución:

- **Alta Cohesión:** patrón que se usa para medir el grado en que están relacionadas las responsabilidades de una clase. Propone asignar responsabilidades de manera que la cohesión se mantenga alta. Este patrón se evidencia en la relación de los *views* con los *models*.
- **Creador:** permite crear objetos de una determinada clase. Es empleado en la mayoría de las clases controladoras para crear instancias de formularios y entidades, es decir, está presente en todos los accesos a datos, ejemplo de ellos en los *models.py*.

- **Controlador:** funciona como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que recibe los datos y las envía a las distintas clases según el método llamado. Se evidencia en la clase `view.py` del *framework* utilizado.
- **Experto:** principio básico de asignación de responsabilidades. Indica que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearla. Se evidencia en la clase `views.py`.

2.6 Modelo de datos

El diseño del modelo de datos consiste en describir la estructura de la base de datos con la representación de la porción que se quiere almacenar en la base de datos. A continuación, en la Ilustración 17, se muestra el diagrama entidad-relación:

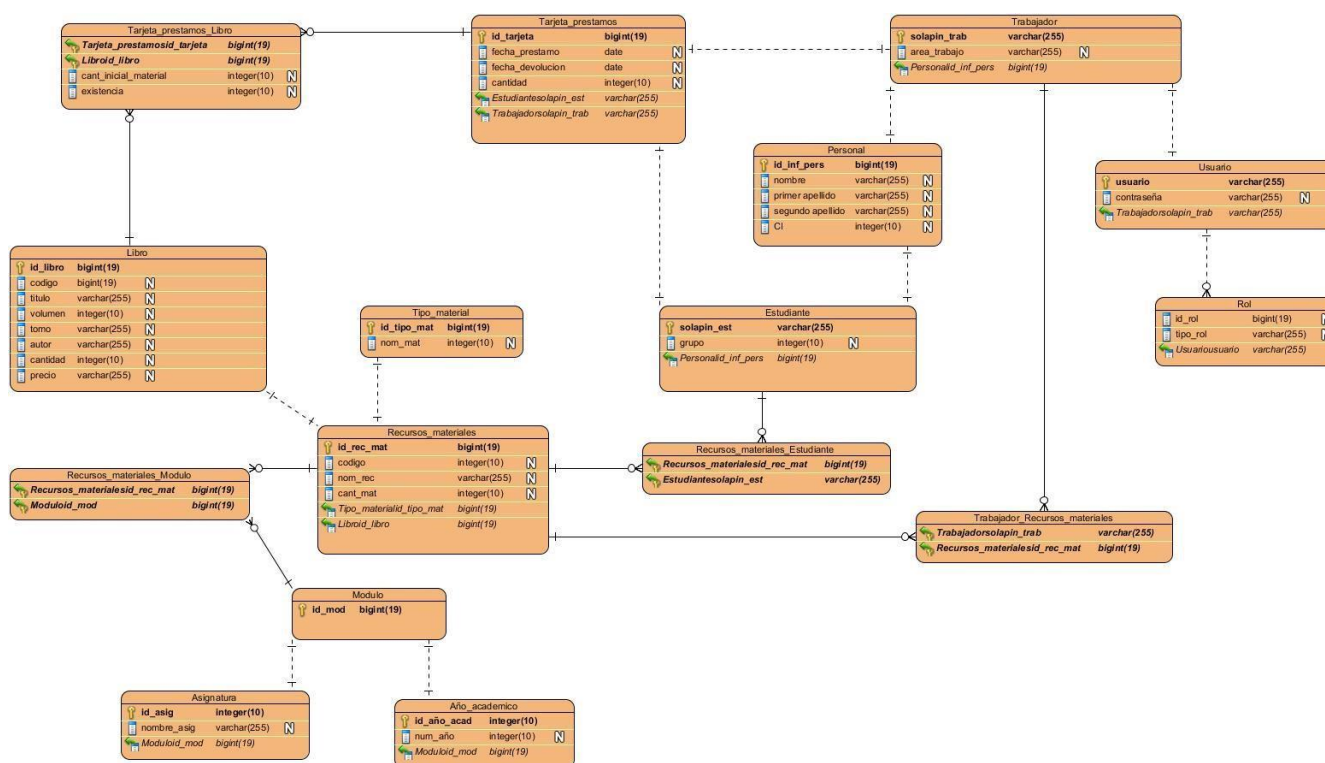


Ilustración 17: Modelo Entidad-Relación (Elaboración propia).

2.6.1 Descripción de las tablas

A continuación, se representa la descripción de la tabla Libro, las demás tablas se encuentran en el Anexo D:

Tabla 7: Descripción de Libro.

Libro		
Descripción: esta tabla contiene la descripción de los atributos de libro.		
Atributos	Tipo	Descripción
id_libro	bigint (19)	Etiqueta única que identifica el libro.
codigo	bigint (19)	Almacena el código del libro.
titulo	varchar (255)	Almacena el título del libro.
volumen	integer (10)	Almacena el volumen del libro.
tomo	varchar (255)	Almacena el tomo del libro.
autor	varchar (255)	Almacena el nombre del autor del libro.
cantidad	integer (10)	Almacena la cantidad existente del libro.
precio	varchar (255)	Almacena el precio del libro.

2.7 Conclusiones del capítulo

Teniendo en cuenta la investigación realizada en el capítulo, se pudo obtener una propuesta de solución. Se identificaron los requisitos funcionales y no funcionales, y se describieron los procesos a través de los DRP, en correspondencia con lo establecido en el escenario 3 de la metodología de desarrollo AUP-UCI, lo que posibilitó una mejor comprensión de los resultados que se quieren obtener. Además, se definieron los patrones de diseño y el patrón arquitectónico MTV, incorporado a *Django, framework* que se va a utilizar en la implementación de la solución. También, se elaboró el diseño de la base de datos que se encargará de guardar toda la información relacionada con la asignación y control de los recursos.

CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBA DEL SISTEMA

En este capítulo se modelan los diagramas de componente y de despliegue, en los que se describen los elementos físicos del sistema con sus relaciones y se representa el comportamiento físico de los componentes de software en los procesadores a través de nodos, respectivamente. Además, se incluyen los tipos de pruebas y validaciones que se realizarán al sistema, con el objetivo de verificar si cumple con los requerimientos establecidos anteriormente.

3.1 Disciplina de implementación

Esta disciplina tiene como objetivo principal desarrollar la arquitectura y el sistema como un todo. Además, se implementan los elementos del Modelo de diseño en términos de componente: código fuente, bibliotecas, script, base de datos. También, estos componentes se representan y organizan en los diagramas de despliegue y de componente (41).

3.1.1 Diagrama de componente

El diagrama de componente describe los elementos físicos del sistema y las relaciones entre ellos. Representa las dependencias entre los componentes del software, en el que se incluyen los componentes de código fuente y ejecutables (41). A continuación, en la Ilustración 18, se muestra el diagrama de componente de la propuesta de solución:

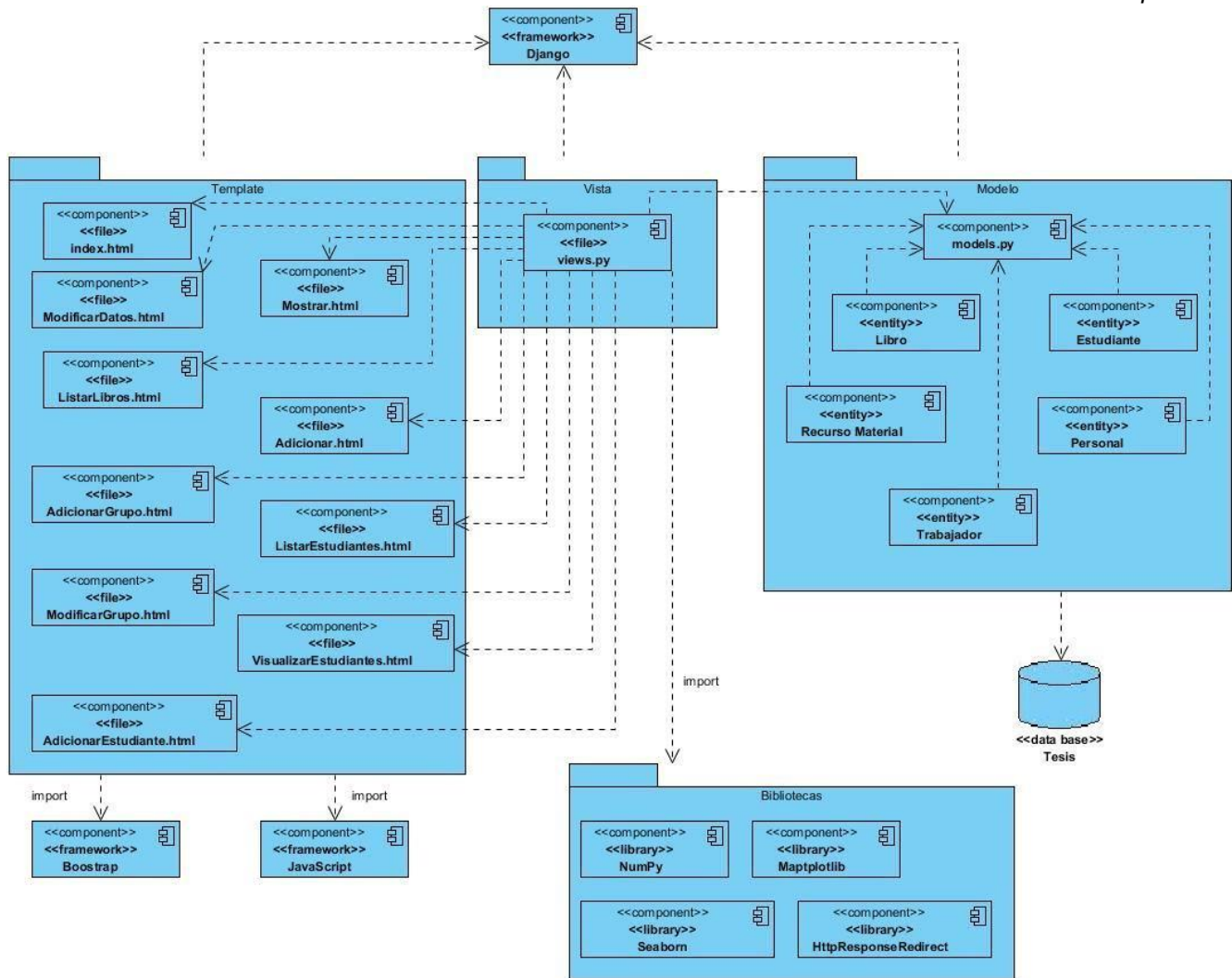


Ilustración 18: Diagrama de Componente (Elaboración propia).

3.1.2 Diagrama de Despliegue

En los diagramas de despliegue es representado el comportamiento físico de los componentes de software en los procesadores, mostrándose el hardware y el software en el sistema, así como el *middleware* usado para conectar los diferentes componentes en el sistema (34). Estos componentes se muestran a través del recurso de ejecución: nodo, en el que su estereotipo permite precisar la naturaleza del equipo: Dispositivo, Procesador y Memoria (38). A continuación, en la Ilustración 19, se muestra el diagrama de despliegue:

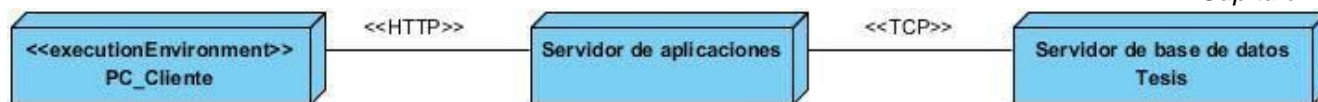


Ilustración 19: Diagrama de Despliegue (Elaboración propia).

3.2 Estándares de codificación

Para la propuesta de solución se decide utilizar los siguientes estándares de codificación:

- Reglas de indentación: utiliza cuatro espacios por indentación y ocho espacios en caso de que sea un código antiguo. Las líneas de continuación deben alinear los elementos ajustados utilizando la línea implícita de *Python* que se une entre paréntesis, corchetes y llaves, o utilizando una indentación en todas las líneas menos la primera. Al usar el último método se debe tener en cuenta, la utilización de una indentación adicional para distinguirse como una línea de continuación.
- Importaciones: las importaciones deben estar generalmente en líneas separadas. Se colocan en la parte superior del archivo. Son agrupadas en el orden siguiente: biblioteca estándar de *Python*, importaciones de terceros y por último importaciones específicas de aplicaciones y/o bibliotecas locales.
- Comentarios: los comentarios pueden ser:
 - 1.3 Comentarios en líneas: se encuentran en la misma línea de una sentencia y deben estar separados por al menos dos espacios de la misma. Comienzan con un `#` y seguidamente de un espacio.
 - 2.3 Bloque de comentarios: se aplican a códigos que lo siguen, y están indentados al mismo nivel que el código. Cada línea comienza con `#` y un solo espacio. Los párrafos dentro de un comentario en bloque están separados por una línea que contiene solamente un `#`.
- Operadores binarios: rodear los operadores binarios con un espacio en cada lado: asignación (`=`), comparaciones (`==`, `<>`, `<=`, `>=`, entre otros), *Booleans* (`and`, `or`, `not`) y asignación de aumento (`+=`, `-=`, entre otros).
- Siempre usar una sentencia `def` en lugar de una sentencia de asignación que une una expresión lambda directamente a un nombre: `def f(x): return 2*x`

- Tabulaciones o espacios: el método de indentación más popular en *Python* es con espacios y el segundo es con tabulaciones, sin mezclar unos con otros.
- Máxima longitud de las líneas: limitar todas las líneas a un máximo de 79 caracteres.
- El código debe estar escrito de tal manera que no provoque inconvenientes con otras implementaciones de *Python* (*PyPy*, *Jython*, *Cython*, etc).

3.3 Pruebas de Software

Las pruebas de software constituyen una herramienta importante para asegurar el correcto cumplimiento de la funcionalidad del producto. Permiten ganar confianza, confirmar la fiabilidad del uso y prevenir defectos en producción (42).

Según el profesor de Ingeniería en el instituto tecnológico de Florida, Cem Kaner, “Las pruebas de software son la investigación empírica y técnica realizada para facilitar a los interesados información sobre la calidad del producto o servicio bajo pruebas” (43). Además, Edsger W. Dijkstra, científico de la computación, define el proceso de pruebas como: “Las pruebas de software pueden ser una manera muy eficaz de mostrar la presencia de errores, pero son totalmente inadecuadas para mostrar su ausencia” (43).

Estas pruebas que son aplicadas en las diferentes etapas del proceso de desarrollo son agrupadas en niveles para verificar y validar el software. A continuación, se muestran los diferentes niveles:

- **Nivel de unidad:** pruebas unitarias o de componente que consisten en la verificación de unidades de software de forma aislada. Además de realizar pruebas de la estructura del código, también se generan casos de pruebas funcionales para comprobar el funcionamiento del componente (43).
- **Nivel de integración:** pruebas de integración que permite probar un conjunto de componentes o subsistemas funcionales del sistema para verificar que interactúan de forma correcta y que se ajustan a los requerimientos funcionales o no funcionales (44).
- **Nivel de sistema:** pruebas del sistema para comprobar si el producto cumple con los requerimientos especificados.

- **Nivel de aceptación:** se realizan las pruebas de aceptación o validación bajo la responsabilidad del cliente y se deben de llevar a cabo antes de que el sistema se ponga en funcionamiento real. Además, debe de satisfacer las expectativas del cliente

(43).

Además, existen técnicas que establecen el grado de profundidad en el que se diseñarán, ejecutarán y evaluarán, todas ellas teniendo como referencia la estructura interna de la aplicación. Estas técnicas son: Caja Blanca y Caja Negra (45).

En la presente investigación se definen las posibles pruebas a realizar en los diferentes momentos del proceso de desarrollo del software. En el nivel de unidad se realizarán pruebas funcionales, aplicando específicamente el método de caja blanca a través de la técnica de camino básico. En el nivel de integración se efectuarán pruebas de integración mediante el *Bottom-up*. Se empleará el método de caja negra, específicamente a través de la partición de equivalencia, en el caso del nivel de sistema. En el último nivel, nivel de Aceptación, se realizarán pruebas de aceptación de tipo alfa. Además, se llevarán a cabo pruebas de confirmación y regresión en todos los niveles para comprobar que las no conformidades que se encuentren sean corregidas y que al hacerlo no se produzcan otros errores. A continuación, se describen cada una de estas pruebas a través de los principales métodos y técnicas empleados y los resultados obtenidos en cada una de estas.

3.3.1 Nivel de unidad

En este nivel se realizan pruebas funcionales, que no son más que pruebas basadas en funciones, presentaciones y en su interoperabilidad con sistemas específicos, y pueden llevarse a cabo en todos los niveles (46).

Método de caja blanca

Las pruebas de caja blanca se basan en el minucioso examen de los detalles procedimentales. Se proponen los casos de pruebas que ejerciten conjuntos específicos de condiciones, a través de la comprobación de caminos lógicos del software (47). Entre las técnicas de pruebas que existen en caja blanca se encuentran: las pruebas de camino básico, pruebas de condición, pruebas de flujo de datos y pruebas de bucles (43).

En la investigación se utiliza la técnica de camino básico, que consiste en verificar el código del sistema de forma tal que se compruebe que todo funcione correctamente. Con esta técnica se intenta garantizar que se prueben que todas las instrucciones del programa se ejecuten, aunque sea una vez (48).

Para desarrollar la prueba de camino básico es posible aplicar los siguientes pasos (48):

- Dibujar el grafo de flujo.
- Calcular la complejidad Ciclomática.
- Determinar el conjunto básico de caminos independientes.

A continuación, se muestra un ejemplo de cómo determinar el mayor de tres valores dados (48):

Paso 1: Dibujar diagrama de flujo

Se detectan los nodos que conformaran al grafo de flujo, así como los caminos que se pueden recorrer durante la ejecución del programa (ver Ilustración 20).

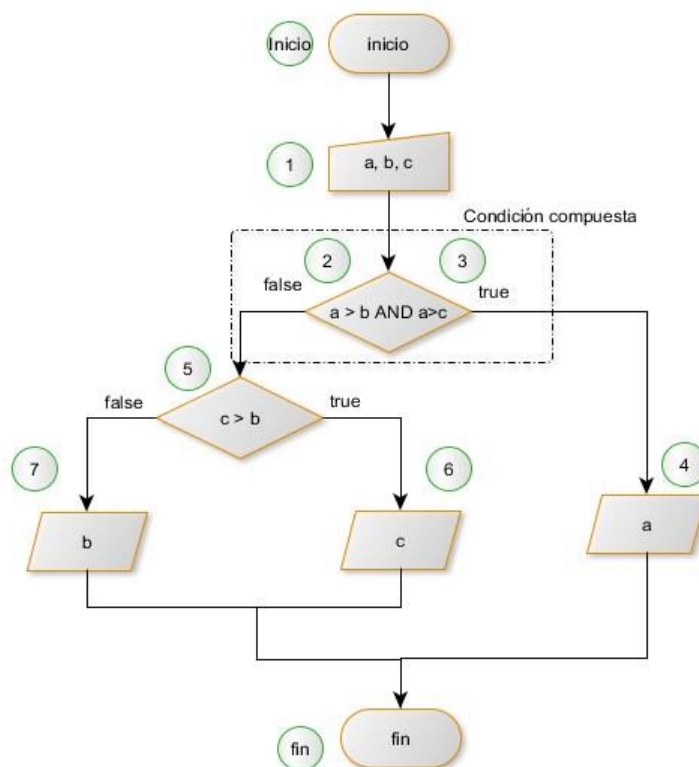


Ilustración 20: Nodos y caminos a recorrer (48).

Al tener una condición compuesta ($a > b$ AND $a > c$), se debe descomponer creando un nodo para cada una de las condiciones. A continuación, en la Ilustración 21, se muestra el grafo de flujo:

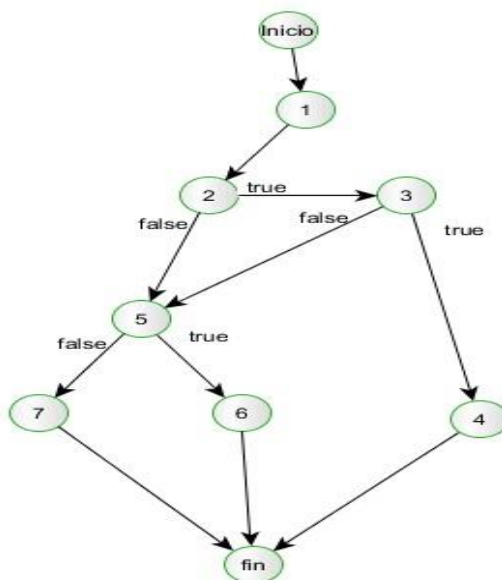


Ilustración 21: Grafo de flujo (48).

Paso 2: Complejidad Ciclomática

La complejidad Ciclomática mide el número de caminos independientes dentro del código que se encuentra en prueba. Para realizar el cálculo se utiliza la fórmula: $V(G) = a - n + 2$

En donde:

a: número de aristas o lados. n:

número de nodos o vértices.

Teniendo en cuenta los datos del ejemplo la formula queda de la siguiente forma:

$$V(G) = 11 - 9 + 2 = 4$$

El código tiene una complejidad ciclomática de 4, por lo que se van a realizar 4 pruebas para asegurar que cada instrucción se ejecute al menos una vez.

Paso 3: Caminos independientes

Los caminos independientes se irán formando desde el más largo al más corto teniendo en cuenta el grafo de flujo (ver Ilustración 22).

CAMINO	ENTRADA	PRUEBA	SALIDA
1,2,3,5,6,F	a>b=TRUE, a>c=FALSE, b>c=TRUE	a=5 b=3 c=7	c
1,2,3,4,F	a>b=TRUE, a>c=TRUE	a=5 b=3 c=4	a
1,2,5,7,F	a>b=FALSE, b>c=FALSE	a=5 b=7 c=6	b
1,2,5,6,F	a>b=FALSE, b>c=TRUE	a=5 b=7 c=9	c

Ilustración 22: Caminos independientes (48).

Teniendo resultados satisfactorios en cada uno de los caminos realizados.

3.3.2 Nivel de integración

Las pruebas de integración son las encargadas de probar las interfaces entre los componentes, las interacciones con diferentes partes de un mismo sistema, como el sistema operativo, el sistema de archivos, el hardware y las interfaces entre varios sistemas (43). Los tres tipos de más comunes de realizarse este tipo de pruebas son:

- *Big bang*: se acoplan todos los módulos de una sola vez, reduciendo la cantidad de pruebas.
- *Bottom-up*: primero se prueban los componentes de bajo nivel y posteriormente se procede hacia los de mayor nivel.
- *Top-down*: primero se prueban los componentes de nivel más alto y luego los de menor nivel.

En la investigación se utiliza el método de *Bottom-up* durante el desarrollo del software.

3.3.3 Nivel del sistema

Con el objetivo de verificar el cumplimiento de los requisitos funcionales establecidos con anterioridad se hace uso del método de caja negra. Este método se refiere a las pruebas que se llevan a cabo en la interfaz del software. Además, utiliza el análisis de la especificación, ya sea funcional o no funcional, sin tener en cuenta la estructura interna del sistema para diseñar los casos de pruebas (43). Entre las técnicas de pruebas existentes en el método de caja negra se encuentran: partición de equivalencia, análisis de valores límites y grafos de causa-efecto.

En la investigación se utiliza el método de caja negra teniendo en cuenta la técnica de partición por equivalencia. Esta técnica permite examinar los valores de las entradas en el software, ya sean validos e inválidos, de los que pueden derivarse casos de pruebas. En la tabla 8, se muestra un ejemplo de los casos de pruebas diseñados y en el Anexo E se encuentran otros diseños de casos de pruebas.

Tabla 8. CP1 Adicionar Libro (Elaboración propia).

Descripción general Permitir agregar un nuevo libro.			
Condiciones de ejecución Para agregar un nuevo libro hay que: Estar autenticado en el sistema con el rol de Encargado del almacén.			
SC 2 Eliminar libro			
Escenario	Descripción	Respuesta del sistema	Flujo Central
EC 1.1 Opción de adicionar libro.	Selecciona la opción de adicionar nuevo libro.	Brinda la posibilidad de introducir un nuevo libro, mostrando los siguientes datos: - Título -Autor Código -Volumen -Tomo -Cantidad -Precio. Además, permitirá: - Guardar -Cancelar.	Administración/Listado de libros/Agregar.
EC 1.2 Opción de guardar los datos	Introduce los datos del libro y selecciona la opción Guardar.	Valida los datos. Crea un libro. Muestra el listado de libros y un mensaje de información.	Administración/Listado de libros/Agregar/Aceptar.
EC 1.3 Opción de cancelar	Selecciona la opción Cancelar.	Elimina los datos creados. Regresa al listado de libros y muestra un mensaje de información.	Administración/Listado de libros/Agregar/Cancelar.
EC 1.4 Datos incompletos	Existen datos incompletos	Muestra un mensaje de información. Muestra un indicador de los campos vacíos: “(*) campos obligatorios”. Regresa al EC 1.1.	Administración/Listado de libros/Agregar/Guardar.

EC 1.5 Datos incorrectos	Existen datos incorrectos.	Muestra un mensaje de información. Muestra un indicador de los campos que están incorrectos.	Administración/Listado de libros/Agregar/Guardar.
		Regresa al EC 1.1.	

3.3.4 Nivel de aceptación

Las pruebas de aceptación se realizan una vez que el sistema se ha implementado en su entorno real de funcionamiento y tiene como objetivo demostrar al cliente que el sistema satisface sus necesidades (47). Estas pruebas a menudo se realizan en dos etapas:

- Pruebas alfa: se lleva a cabo en el lugar de desarrollo del producto por el cliente o personal que representa al cliente. El desarrollador estará junto a los usuarios registrando los errores y problemas de uso.
- Pruebas beta: se desarrollan en el entorno del cliente, por lo que el desarrollador no se encontrará presente. El cliente registrará los fallos derivados del uso del producto para posteriormente informarle al desarrollador.

La validación del producto de software dentro de la estrategia de pruebas definida, se planifica con la variante de pruebas de aceptación alfa. Esta constituye una variante viable debido a que el cliente ha interactuado en el entorno de desarrollo, conociendo de esta manera, las condiciones bajo las cuales se realizarán las pruebas.

3.3.5 Pruebas de regresión y confirmación

Las pruebas de regresión son una estrategia de prueba en la cual las pruebas que se han realizado anteriormente en el programa se vuelven a efectuar en la nueva versión, para asegurar la calidad del producto luego de haber añadido las nuevas funcionalidades. Además, comprueba que se hayan corregido los defectos identificados en la prueba anterior y que las modificaciones efectuadas no produzcan efectos negativos sobre el mismo u otros componentes. Normalmente, este tipo de pruebas implica la repetición de las pruebas que se han realizado previamente, por lo que se llevan a cabo durante cada iteración.

También, están las pruebas de confirmación, que son las pruebas que se realizan para comprobar si se corrigieron o no las no conformidades identificadas y verificar si surgieron nuevos errores. Además, al igual que las pruebas de regresión, estas se aplicarán en cada uno de los niveles.

3.4 Validación

Para validar el sistema se hará uso de la técnica IADOV, que está compuesta por cinco preguntas: tres cerradas y dos abiertas. Además, constituye una vía indirecta para el estudio de la satisfacción, debido a que los criterios que utiliza se fundamentan en las relaciones que se establecen entre las tres preguntas cerradas que son intercaladas dentro de un cuestionario y cuya relación el sujeto desconoce.

Las preguntas se relacionan a través del cuadro lógico de IADOV y el número restante de la interrelación de las tres preguntas, indica la posición de los sujetos en la escala de satisfacción.

La escala de satisfacción está dada por los siguientes criterios (49):

- Máxima satisfacción.
- Más satisfecho que insatisfecho.
- No definida.
- Más insatisfecho que satisfecho.
- Máxima insatisfacción.
- Contradictoria.

Tabla 9: Cuadro lógico de IADOV (Elaboración propia).

Tabla 6. Cuadro lógico de INDOV (Elaboración propia).									
	¿Considera usted que el sistema desarrollado permite que la asignación y control de recursos se realice de forma más eficiente?								
	No			No Sé			Si		
¿Considera usted que el sistema desarrollado, da solución a todos los problemas identificados en los procesos de asignación y control de recursos?	¿Considera usted factible el desarrollo de un sistema que permita la asignación y control de recursos de la facultad 4?								
	Si	No sé	No	Si	No sé	No	Si	No sé	No
Me gusta mucho									
No me gusta mucho									

Me da lo mismo									
Me disgusta más de lo que me gusta									
No me gusta nada									
No sé qué decir									

Para obtener el índice de satisfacción grupal (ISG) se trabaja con los diferentes niveles de satisfacción expresados en la escala numérica que oscila entre +1 y -1.

Tabla 10: Índice de satisfacción grupal (49).

Índice de satisfacción	Escala
Máxima satisfacción	+1
Más satisfecho que insatisfecho	+0.5
No definido y contradictoria	0
Más insatisfecho que satisfecho	-0.5
Máxima insatisfacción	-1

Para calcular la satisfacción grupal se calcula a través de la siguiente formula:

$$ISG = \frac{A+B+C+D+E}{N}$$

En donde A, B, C, D, E representan el número de sujetos con índices individual 1; 2; 3 o 6; 4; 5 y N representa el número total de sujetos.

3.5 Conclusiones del capítulo

Se generó el diagrama de componente para entender la relación de dependencia y uso de los componentes. Los estilos y estándares de codificación permitieron organizar el código de la propuesta de solución. Las pruebas de softwares permitirán identificar errores en el producto donde a través de: la aplicación del método de caja blanca con la utilización de la técnica de camino básico permitirá comprobar el funcionamiento satisfactorio de los métodos implementados; el método de caja negra con la técnica de partición de equivalencia y la utilización de los diseños de casos de pruebas proporcionará la corrección de las no conformidades que son identificadas, garantizando un nivel alto de satisfacción y calidad.

CONCLUSIONES FINALES

A partir de la investigación realizada se logra arribar a las siguientes conclusiones generales:

- El estudio de sistemas informáticos basados en tecnologías web, permitió identificar funcionalidades que se encuentran asociadas a la asignación y control de recursos, y que pueden formar parte de la propuesta de solución.
- La confección de los principales artefactos correspondientes al flujo de trabajo de la metodología de desarrollo software AUP-UCI en su escenario 3, posibilitó la descripción de las necesidades dentro del proceso de desarrollo de la solución y las precisiones de las bases para la ejecución de etapas posteriores en el ciclo de vida del sistema.
- El análisis y diseño del software, a partir de los resultados obtenidos en la fase de requerimientos, permitió definir la estructura y el comportamiento del sistema que se va a desarrollar.
- Las pruebas de software descritas permitirán identificar y corregir los errores del sistema, evitando insatisfacciones por parte del cliente, demostrándose de esta forma el cumplimiento de los objetivos planteados en la investigación.

Recomendaciones

RECOMENDACIONES

Para asegurar la continuidad de la investigación realizada, es recomendado:

- Culminar con la implementación del sistema.
- Realizar las diferentes pruebas de software que fueron diseñadas y que no fueron posible aplicar en el sistema.

REFERENCIAS BIBLIOGRÁFICAS

1. Riascos, E., Aguilera, C. y Solano, R. *Efectividad de las TIC en los procesos administrativos*. Santiago de Cali - Colombia : Gerenc. Tecnol. Inform., 2015. Vol. 14.
2. *Ventajas del uso de las TICs en el proceso de enseñanza-aprendizaje desde la óptica de los docentes universitarios españoles*. Soto, C. F., Senra, A. I. y Neira, M. C. 2009, Edutec. Revista Electrónica De Tecnología Educativa, (29).
3. Salmerón, Alejandro. *sitio web de MEDAC*. [En línea] [Citado el: 11 de Junio de 2021.] <https://medac.es/blogs/sociocultural/las-herramientas-tic-en-la-educacion>.
4. Solano, Fernando. [En línea] 29 de Julio de 2019. [Citado el: 13 de Junio de 2021.] <https://blog.imagineer.co/es/que-es-un-proceso-de-negocio>.
5. Oro, Fidel Esteban. *Propuesta de las tecnologías de información y comunicación (TIC's) en el apoyo administrativo*. Lima : s.n., 2019.
6. Padilla, Lillian. *Planeación de llos recursos de la empresa: ERP*.
7. Fernando. *asesorias.com*. [En línea] 2020. [Citado el: 13 de Junio de 2021.] <https://asesorias.com/empresas/programas-gratis/software-gestion-almacenes/>.
8. SoftExpert. *SoftExpert Excelent Suite*. [En línea] 2020. [Citado el: 13 de Junio de 2021.] <https://www.softexpert.com/es/produoto/gestion-almacenes/>.
9. opengeekservice. *open geek*. [En línea] 2015. [Citado el: 13 de Junio de 2021.] <https://www.opengeekservice.cl/sitio/es/blog/62-koha-el-software-open-source-para-laautomatizaci%C3%B3n-de-bibliotecas.html>.
10. ABCdatos. *ABCdatos*. [En línea] 2019.
11. *sitio web Ahora*. [En línea] <https://www.ahora.es/productos/sga/>.
12. Tienda Digital. [En línea] 2020. <https://tienda.digital/los-top-50-sofware-de-planificacion-derecursos-empresariales-erp-globales-para-pequenas-y-medianas-empresas-pymes/>.
13. Maida, Esteban G. y Pacienza, Julián. *Metodologías de desarrollo de software*. Facultad de Química e Ingeniería "Fray Rogelio Bacon", Universidad Católica Argentina. 2015. Tesis de Licenciatura en Sistemas y Computación.
14. Cruz, Nancy y González, Antonio. Tradicional o ágil? La metodología ágil como alternativa a la transformación. *sitio web de Axpe Consulting*. [En línea] 2018.
15. Rodríguez Sánchez, Tamara. *Metodologías de desarrollo para la actividad productiva de la UCI*.
16. Rumbaugh, James, Jacobson, Ivar y Booch, Grady. *Unified Modeling language*. 2004.
17. García, Tito y Rubí, Sandra. *UML Introduccion al UML, modelando con UML, utilidad del UML, conceptos de USE CSE, objetos, clases y atributos, operaciones, aplicaciones*. Lima : s.n., 2018.
18. Díaz, Eduardo, y otros. *Generación de interfaces de usuario a partir de modelos BPMN con estereotipos*. 2018.
19. *Sitio oficial de la herramienta Visual Paradigm*. [En línea] [Citado el: 25 de Junio de 2021.] <http://www.visual-paradigm.com/>.
20. Luzuriaga Mendoza, Amanda Maribel. *Análisis de factores que inciden en la selección de un lenguaje y framework de programación para desarrollo de software web*. Facultad de Ingeniería Civil, Universidad Técnica de Machala. Machala : s.n., 2020.
21. al., N. R. Swain et. *A review of open source software solutions for developing*. 2015.

22. Palomo, Manuel y Montero, Ildelfonso. *Programación en PHP a través de ejemplos*. Departamento de Lenguajes y Sistemas Informáticos.
23. ROSEPAC. Java vs Python, ¿Cuáles son sus diferencias? *Ciberninjas*. [En línea] 28 de Mayo de 2020. [Citado el: 23 de Junio de 2021.]
file:///F:/Comparativa%20lenguajes%20de%20programacion/Java%20vs%20Python,%20%C2%BFCu%C3%A1les%20son%20sus%20diferencias_%20-%20Ciberninjas.html.
24. Germain, Cristal E. *Desarrollo de aplicaciones web utilizando JavaScript*. Universidad Politécnica de Sinaloa. Sinaloa : s.n., 2020.
25. Desarrollo Web: Tabla comparativa de los lenguajes de programación. *Desarrollo Web*. [En línea] 2 de Febrero de 2015. [Citado el: 1 de Julio de 2021.]
<http://desarrollowebbydesarrolloweb.blogspot.com/2015/02/tabla-comparativa-de-los-lenguajesde.html>.
26. Vega Villamonte, Daniel Alejandro. *Sistema experto como soporte a la programación con lenguaje Python3.x basado en Microsoft Bot Framework*. Facultad de ingeniería y arquitectura, Universidad Privada Telesup. Lima : s.n., 2018. Tesis.
27. Prasad Acharya, Durga. Kinsta Blog: PHP vs Python: Una Comparación Detallada Entre los Dos Lenguajes . *Sitio Web de Kinsta Inc*. [En línea] 13 de Mayo de 2021. [Citado el: 1 de Julio de 2021.]
28. Molina Rios, Jimmy Rolando, y otros. *Evaluación de frameworks en el desarrollo de aplicaciones web con Python*. Departamento de investigación, Universidad Técnica de Machala. Machala : s.n., 2016.
29. Domingo Muñoz, José. *Sitio Web de OpenWebinars*. [En línea] 17 de Noviembre de 2017. [Citado el: 3 de Julio de 2021.]
30. *Sitio Web de Ciberninjas- El Diario Tecnológico del Futuro*. [En línea] 2021. [Citado el: 3 de Julio de 2021.] <https://ciberninjas.com/python-frameworks/>.
31. *sito web de programmer click*. [En línea] [Citado el: 2 de Julio de 2021.] <https://programmerclick.com/article/62881980004/>.
32. Acero Linares, J. C. Las mejores bases de datos. *CoRegistros*. [En línea] 2018. [Citado el: 2 de Julio de 2021.] <https://www.coregistros.com/mejores-bases-de-datos/>.
33. Yumiseba Sanunga, Piedad Karina y Gutiérrez Mancheno, Christian Rafael. *Desarrollo de un sistema informático para automatizar el registro de pacientes atendidos por el servicio de emergencia del Hospital Pediátrico Alfonso Villagómez aplicando el framework Django*. Facultad de informática y electrónica, Escuela Superior Politécnica de Chimborazo. Riobamba : s.n., 2018. Trabajo de titulación: proyecto técnico.
34. Sommerville, Ian. *Ingeniería de Software*. Novena. 2011. pág. 792.
35. Sarmiento Cuervo, Patricia y Carolina Hernández, Diana. *Metodología para la optimización de los procesos de recolección de información y análisis en la etapa de especificación de requerimientos de software*. Bogotá : s.n., 2017.
36. Tallada Cebrián, Arnau. *Implementación de la tecnología RFID en el transporte público*. 2015.
37. Onofre Avila, Bryan Alain. *Análisis de aplicaciones web utilizando Python para el desarrollo del lado del backend en instituciones públicas*. FACULTAD DE ADMINISTRACIÓN, FINANZAS E INFORMÁTICA, UNIVERSIDAD TÉCNICA DE BABAHOYO. 2021.
38. Larman, C. *UML y Patrones. Introducción al análisis orientado a objetos*. 1999.

39. Suárez Govea, Pedro Pablo. *Componente de revisión de recursos educativos para la plataforma educativa Zera 2.0*. Facultad 4, Universidad de las Ciencias Informáticas. 2016. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.
40. Iturralde, O.J.B. *Introducción a los Patrones de Diseño*. 2016.
41. Jacobson, Ivar. *El Proceso Unificado de Desarrollo de Software*. 2004.
42. Campos Chiu, Cindy. *Las pruebas en el desarrollo de software*. 2015.
43. Sánchez Peño, José Manuel. *Pruebas de Software. Fundamentos y Técnicas*. 2015.
44. Echevarria, Raul Antonio Ramirez. [En línea] 2014. <https://prezi.com/0mpgx-lmytat/pruebas-deintegracion/>.
45. VELÁSQUEZ, C.D.J.C. *PROPUESTA METODOLÓGICA PARA LA REALIZACIÓN DE PRUEBAS DE DE SOFTWARE EN UN AMBIENTES PRODUCTIVOS*. Universidad Nacional de Colombia. 2009.
46. Mera Paz, Julián Andrés. *Análisis del proceso de pruebas de calidad de software*. 2016.
47. Pressman, Roger S. *Ingeniería del Software: Un enfoque práctico*. 2005.
48. Mouse, JC. [En línea] 5 de Octubre de 2017. [Citado el: 4 de Noviembre de 2021.] <http://www.jcmouse.net/ingenieria-de-sistemas/caja-blanca-prueba-del-camino-basico>.
49. Vargas Aguila, Yairon, y otros. *Sistema Integral de Control Interno para el Vicedecanato de Administración y Servicios de la Facultad 3*. Universidad de las Ciencias Informáticas. 2017. Artículo.
50. *Sitio Web de Python Software Foundation*. [En línea] 2021. [Citado el: 7 de Julio de 2021.] <https://docs.python.org/es/3/library/tkinter.html#a-very-quick-look-at-tcl-tk>.
51. *Sitio Web de ArtanSoft*. [En línea] 2017. [Citado el: 7 de Julio de 2021.] <https://artansoft.com/2017/07/vim-ctags-autogenerar-etiquetas/>.

ANEXOS

Anexos A: Descripción de los roles y responsabilidades

Tabla 11: Roles y Responsabilidades (Elaboración propia).

Roles	Responsabilidades
Decana	Máximo responsable del control interno de la facultad.
Vicedecano de Administración y Economía	Máximo responsable de controlar el destino final de todos los recursos de la facultad.
Administrador del docente	Encargado de dar entrada y salida a los insumos de limpieza en el Libro de control de insumos. Es el máximo responsable de que estos recursos lleguen a su destino final correspondiente: Auxiliares Generales de Servicio o a las oficinistas.
Asistente de Control	Responsable de entregar los recursos materiales a los estudiantes y trabajadores. Además, debe mantener actualizada y firmadas las actas de destino final.
Responsable del almacén	Encargado de la entrega, recepción y control de los libros en el almacén.
Trabajadores y estudiantes	Personal de la facultad al que le es entregado los recursos materiales.
Personal de Servicio	personal al que le es entregado los insumos de limpieza.

Anexos B: Modelo de los procesos To-Be

Disciplina del Modelado del negocio:

1. Asignación de recursos

La vicedecana de administración y economía de la facultad debe verificar si es necesario o no solicitar los recursos (ver Ilustración 23). En caso de que no se tenga que realizar una solicitud, la vicedecana confirma directamente la entrega de los materiales a los encargados de realizar la entrega. Este proceso de entrega se divide en tres subprocesos: Entregar recursos materiales, Entregar insumos de limpieza y Entregar de libros a estudiantes o trabajadores, los cuales van hacer supervisados por el asistente de control, el administrador del docente y el responsable del almacén respectivamente. En caso contrario, se realiza el proceso de solicitar materiales, para posteriormente efectuar la entrega de materiales teniendo en cuenta lo antes descrito.

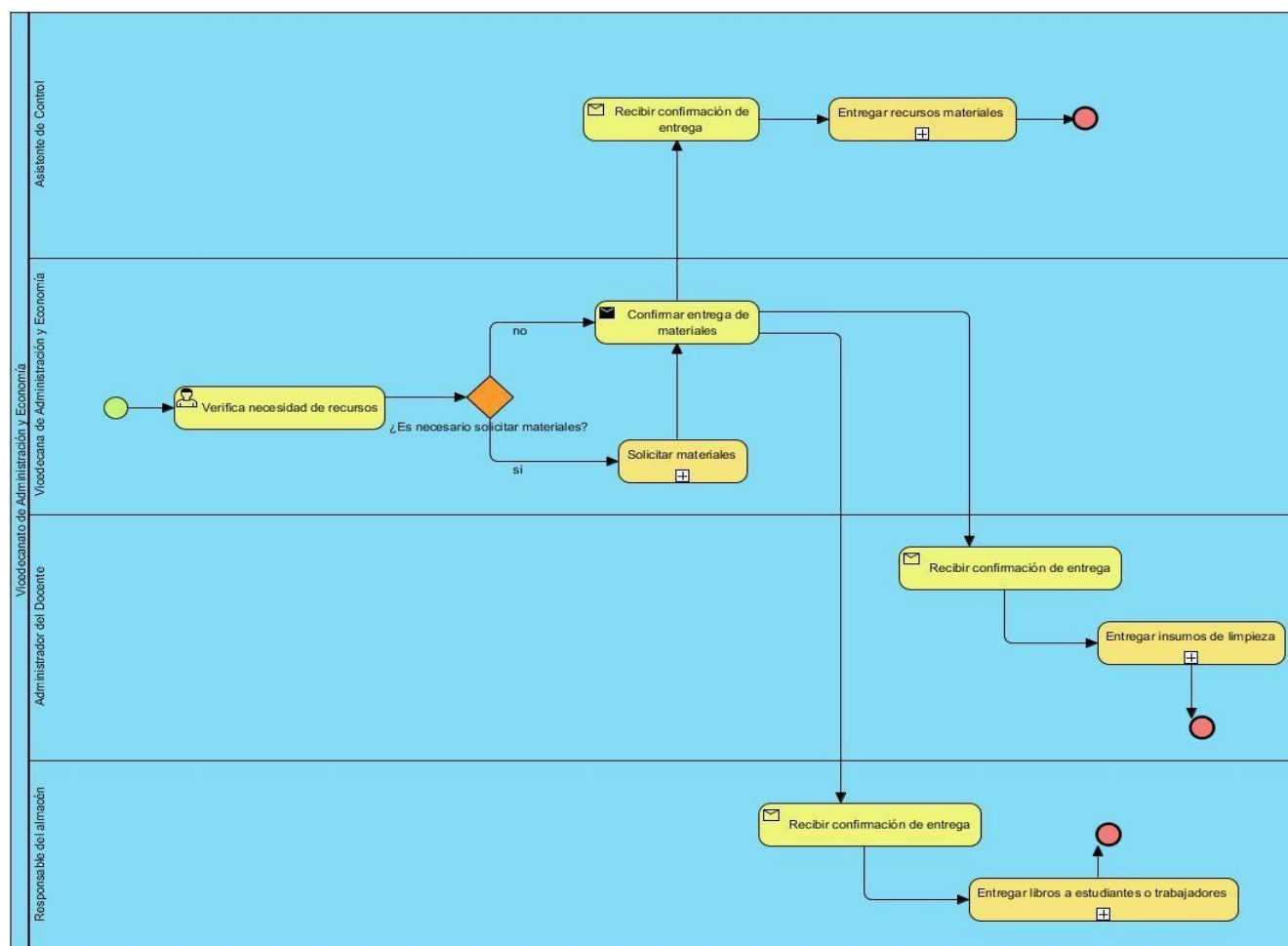


Ilustración 23: Modelo To-Be del Proceso Asignación de recursos (Elaboración propia).

2. Entregar recursos materiales

El asistente de control verifica en el sistema si el solicitante pertenece a la facultad (ver Ilustración 24). De pertenecer, se le hace entrega de los recursos materiales y se le realiza un acta de entrega de estos recursos. Posteriormente se imprime esta acta y se le entrega al solicitante para que la firme y luego se

archiva. En caso contrario, que el solicitante no este registrado en el sistema, no se procede con la entrega de recursos.

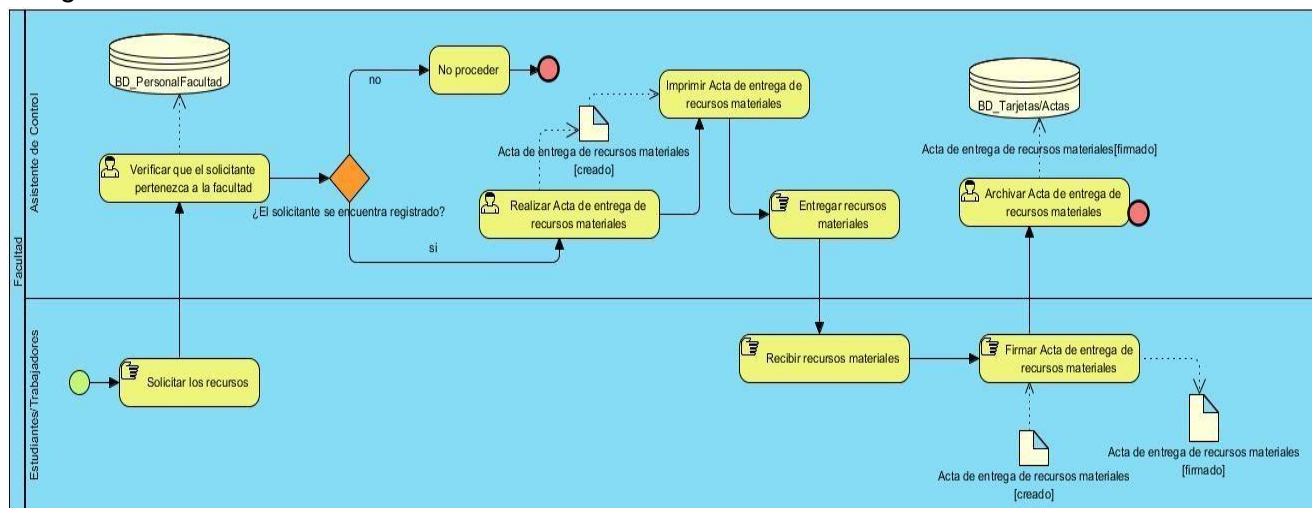


Ilustración 24: Modelo To-Be del proceso Entregar recursos materiales (Elaboración propia).

3. Entregar insumos de limpieza

El Vicedecano de Administración y Economía hace entrega de los insumos de limpieza al administrador del docente, quien es el encargado de registrar los insumos en el Libro de control de insumos (Ver Ilustración 25). Luego procede a entregar estos insumos al personal de servicio y a realizar un acta de entrega que debe firmar el personal. Seguidamente se archiva el acta de entrega.

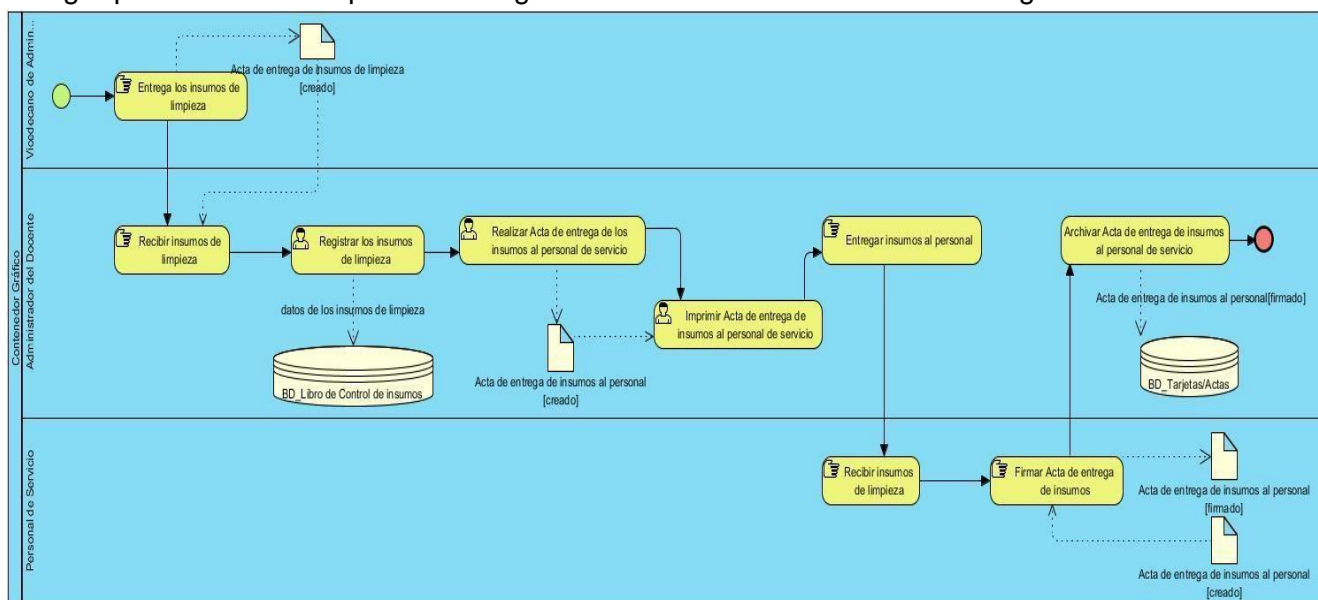


Ilustración 25: Modelo To-Be del proceso Entregar insumos de limpieza (Elaboración propia).

4. Recepcionar libros

El estudiante desea realizar la entrega de los libros que posee (ver Ilustración 26). En este caso el responsable del almacén debe recepcionar dichos libros, para ello debe comprobar su solapín para verificar que pertenezca a la facultad. Luego, busca en el sistema el grupo al que pertenece y por ende

el nombre del estudiante en el listado. Posteriormente, el responsable del almacén debe recepcionar los libros y actualizar la tarjeta de préstamos. En caso de que el estudiante se encuentre en proceso de baja y le falten libros por entregar, el responsable del almacén debe realizar un modelo de débito. Si es un trabajador se procede de igual manera.

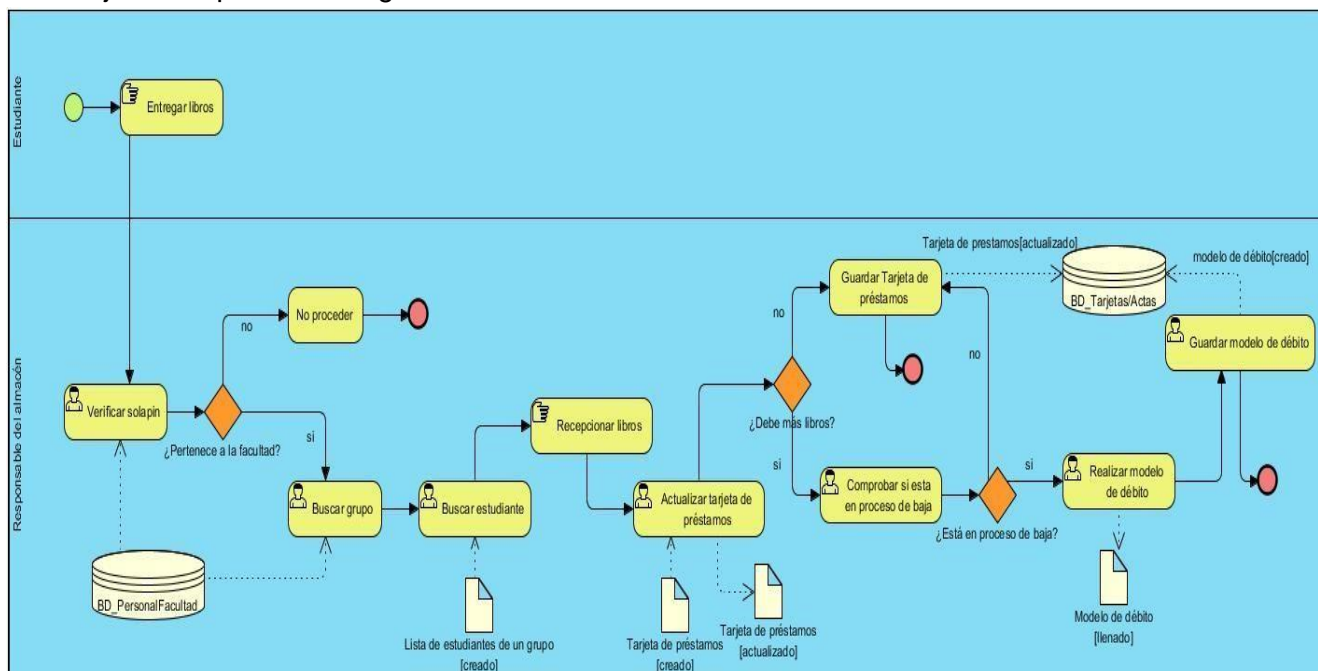


Ilustración 26: Modelo To-Be del proceso Recepcionar libros entregados a los estudiantes (Elaboración propia).

5. Control de recursos materiales

El control de recursos materiales se encuentra dividido en dos, en donde el control de los libros tiene como responsable al jefe del almacén y la parte del control de los recursos materiales y de los insumos de limpieza es responsabilidad del Vicedecano de Administración y Economía de la facultad. Para un mejor entendimiento de este proceso se llevará a cabo dos subprocesos:

Controlar los libros en el almacén

El responsable del almacén realiza un inventario de los libros que tiene en el almacén para luego realizar una comparación con la existencia en la base de datos de los libros (ver Ilustración 27). Para verificar esto, se tiene en cuenta el listado de los libros rotos, los perdidos y los que están entregados. Si al realizar el conteo se tienen diferencias se procede a realizar una investigación para corregir los errores y luego actualizar la base de datos. En caso contrario se actualiza directamente la base de datos.

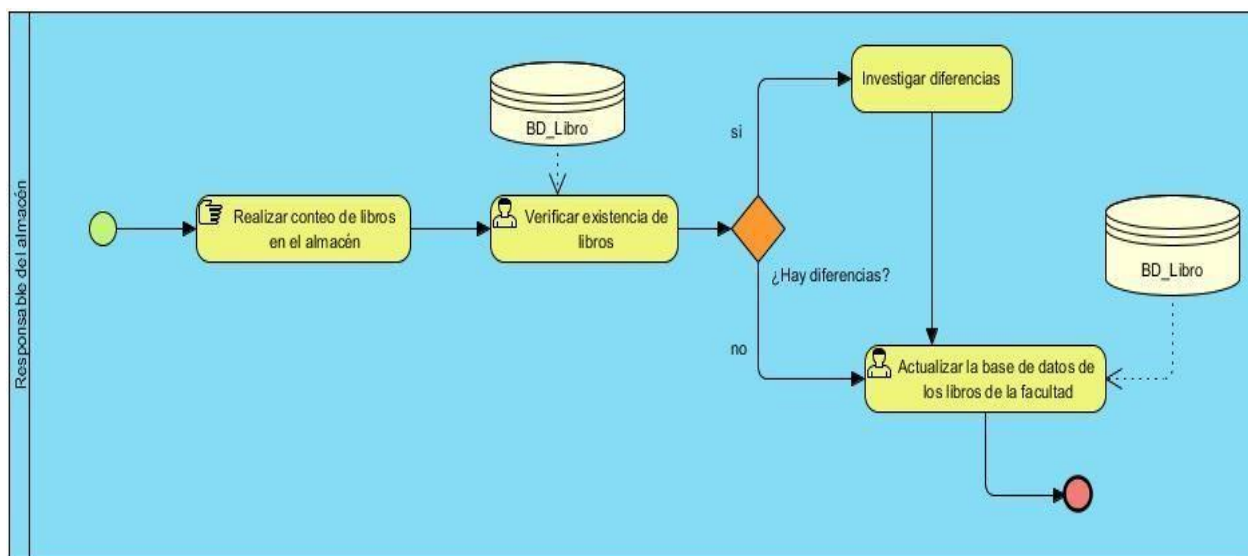


Ilustración 27: Modelo To-Be del proceso Controlar libros en el almacén (Elaboración propia).

Control de recursos

El Vicedecano de Administración y Economía de la facultad realiza un control de los insumos de limpieza o recursos materiales, a través de una comparación con la existencia en la base de datos de los recursos materiales (ver Ilustración 28). Para verificar, se tiene en cuenta, la cantidad de recursos entregados a la facultad (vales de salida) y los que se les entrego al personal (actas de destino final). Si al finalizar el control se tienen irregularidades se procede a efectuar una investigación con los responsables (Administrador del docente o el Asistente de control) para corregir los errores y luego actualizar la base de datos de las actas de destino final. En caso contrario se actualiza directamente la base de datos.

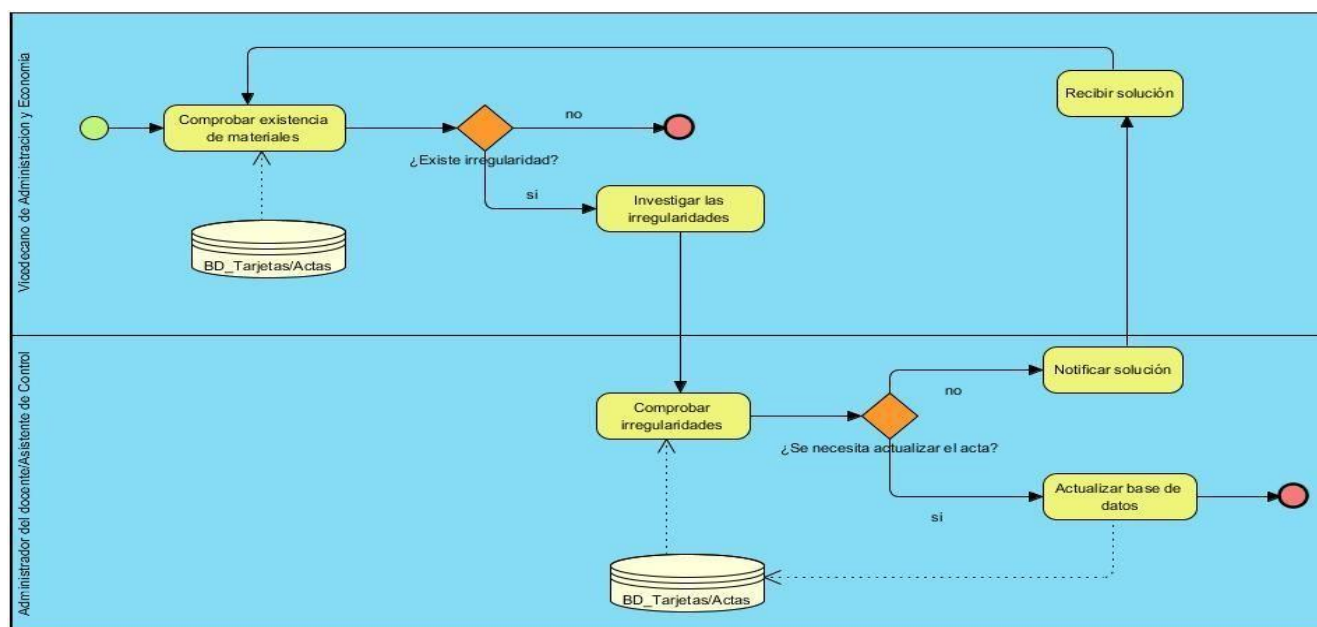


Ilustración 28: Modelo To-Be del proceso Control de recursos (Elaboración propia).

Anexos C: Diagramas de Clases del diseño

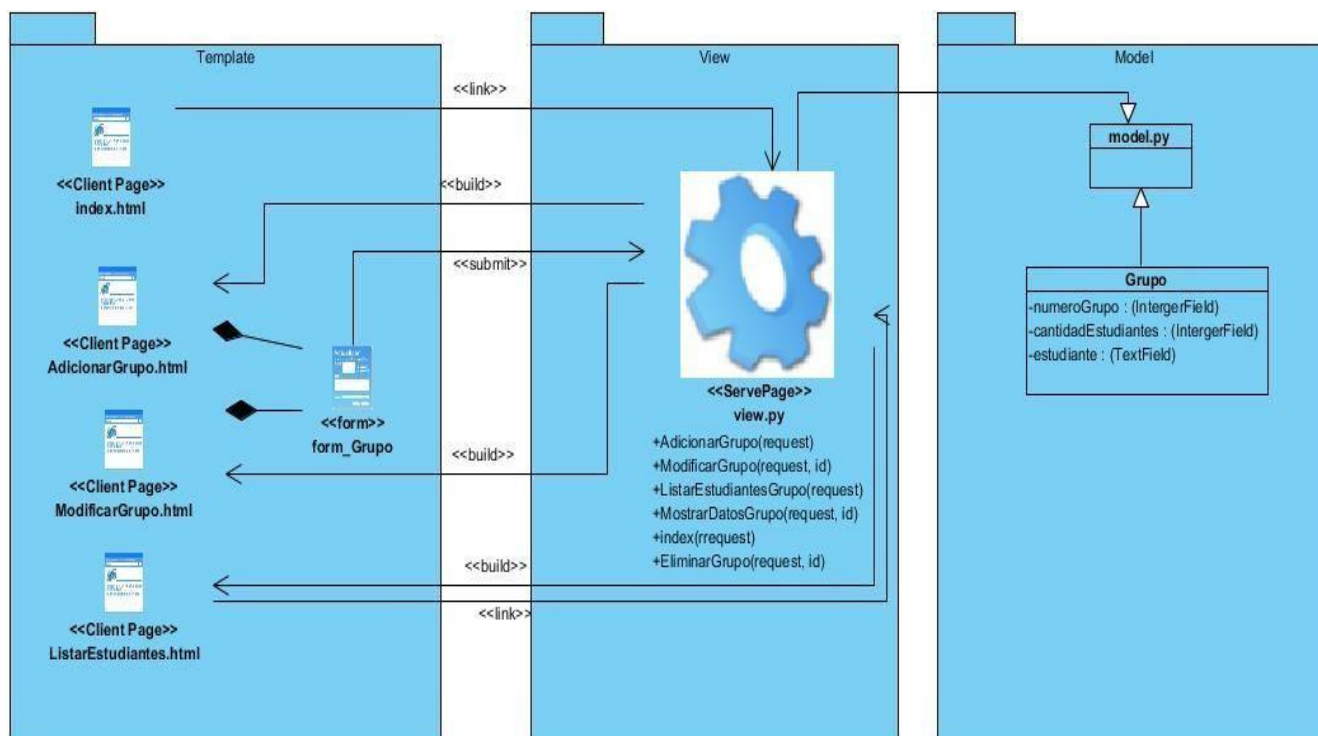


Ilustración 29: Diagrama de Clases del diseño Gestionar Grupo (Elaboración propia).

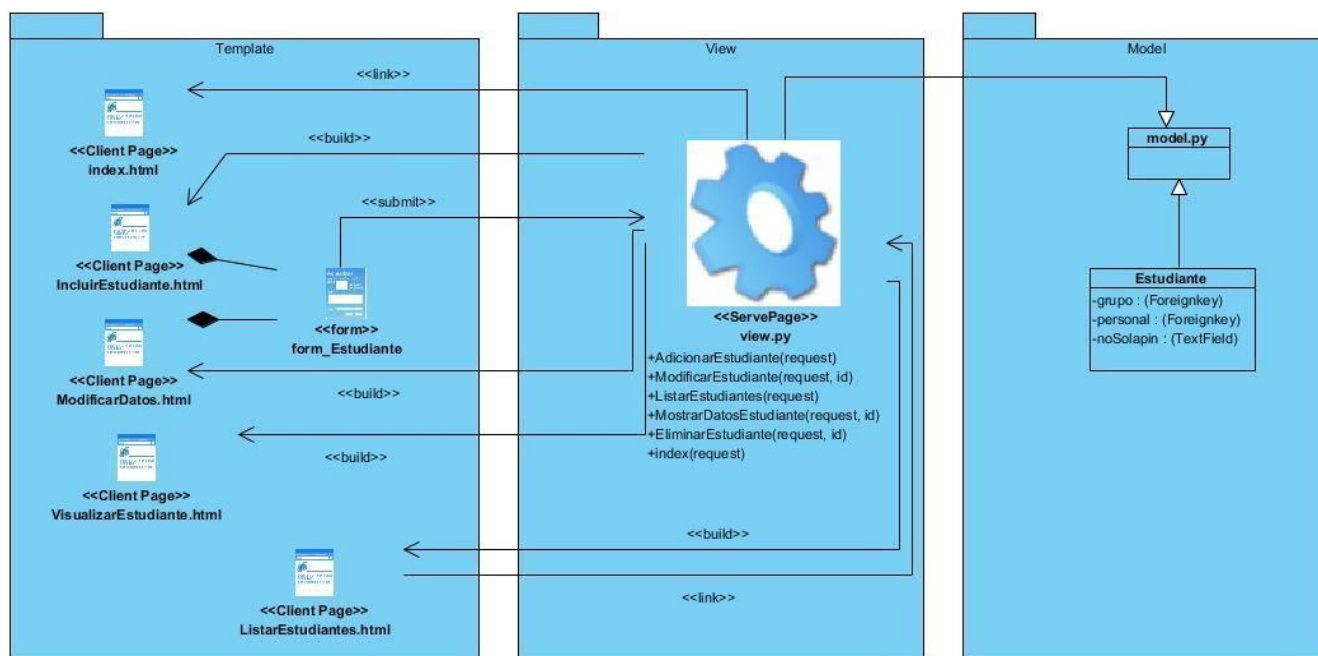


Ilustración 30: Diagrama de Clases del diseño Gestionar Estudiante (Elaboración propia).

Anexos D: Descripción de las tablas de la base de datos

Tabla 12: Descripción de Estudiante (Elaboración propia).

Estudiante		
Descripción: en esta tabla se describen los atributos de los estudiantes.		
Atributos	Tipo	Descripción
solapin_est	varchar (255)	Etiqueta única que identifica al estudiante.
grupo	integer (10)	Almacena la información del grupo docente al que pertenece el estudiante.
id_inf_pers	bigint (19)	Etiqueta que contiene la información personal del estudiante.

Tabla 13: Descripción de Trabajador (Elaboración propia).

Trabajador		
Descripción: en esta tabla se realiza una descripción de los atributos de los trabajadores.		
Atributos	Tipo	Descripción
solapin_trab	varchar (255)	Etiqueta única que identifica al trabajador.
area_trabajo	varchar (255)	Almacena la información del área de trabajo al que pertenece el trabajador.
id_inf_pers	bigint (19)	Etiqueta que contiene la información personal del trabajador.
usuario	varchar (255)	Etiqueta que contiene el usuario perteneciente al trabajador.

Tabla 14: Descripción de Usuario (Elaboración propia).

Usuario		
Descripción: en esta tabla se describen los atributos del usuario.		
Atributos	Tipo	Descripción
usuario	varchar (255)	Etiqueta única que identifica a cada usuario.
contraseña	varchar (255)	Almacena la contraseña

Tabla 15: Descripción de Personal (Elaboración propia).

Personal		
-----------------	--	--

Descripción: esta tabla describe los atributos de la información personal.		
Atributos	Tipo	Descripción
id_inf_pers	bigint (19)	Etiqueta única que identifica a cada persona.
nombre	varchar (255)	Almacena el nombre del personal.
primer apellido	varchar (255)	Almacena el primer apellido del personal.
segundo apellido	varchar (255)	Almacena el segundo apellido del personal.
CI	integer (10)	Almacena el número de carnet de identidad del personal.

Tabla 16: Descripción de Recursos Materiales (Elaboración propia).

Recursos_materiales		
Descripción: en esta tabla se realiza la descripción de los recursos materiales de la facultad.		
Atributo	Tipo	Descripción
id_rec_mat	bigint (19)	Etiqueta única que identifica el recurso material.
id_tipo_material	bigint (19)	Etiqueta que contiene el tipo de material.
código	integer (10)	Almacena el código del recurso material.
cant_mat	integer(10)	Almacena la cantidad del tipo de material.
nom_rec	varchar(255)	Almacena el nombre del material.
id_libro	bigint(19)	Etiqueta que contiene la información de libro.

Tabla 17: Descripción del Tipo de material (Elaboración propia).

Tipo_material		
Descripción: en esta tabla se describen los atributos del tipo de recurso material.		
Atributos	Tipo	Descripción
id_tipo_mat	bigint (19)	Etiqueta única que identifica cada material.
nom_mat	varchar (255)	Almacena el nombre del tipo de material.

Tabla 18: Descripción de Tarjeta de préstamos (Elaboración propia).

Tarjeta_préstamos		
Descripción: en esta tabla se describen los atributos de la tarjeta de préstamos.		
Atributos	Tipo	Descripción
id_tarjeta	bigint (19)	Etiqueta única que identifica cada tarjeta de préstamos.
solapin_est	varchar (255)	Etiqueta que contiene la información del estudiante.
solapin_trab	varchar (255)	Etiqueta que contiene la información del trabajador.
fecha_prestamo	date	Almacena el día en que se realiza la entrega del recurso.
fecha_devolucion	date	Almacena el día en que se efectúa la recepción del recurso.
cantidad	integer (10)	Almacena la cantidad de recursos que se prestó.

Tabla 19: Descripción de la relación de Libro y Tarjeta de préstamos (Elaboración propia).

Tarjeta_préstamos_Libro		
Descripción: en esta tabla se describen los atributos de la relación entre libro y préstamos.		
Atributos	Tipo	Descripción
id_tarjeta	bigint (19)	Etiqueta única que identifica cada tarjeta de préstamos.
cant_inicial_material	integer (10)	Almacena la cantidad inicial de materiales que entran en el almacén.
existencia	integer (10)	Almacena la cantidad de materiales existentes en el almacén.
id_libro	bigint (19)	Etiqueta que contiene la información de libro.

Tabla 20: Descripción de Módulo (Elaboración propia).

Modulo		
Descripción: esta tabla contiene la descripción de los atributos de módulo.		
Atributos	Tipo	Descripción
id_mod	bigint (19)	Etiqueta única que identifica el módulo.

Tabla 21: Descripción de Módulo por asignatura (Elaboración propia).

Asignatura		
Descripción: esta tabla contiene la descripción de los atributos del módulo por asignatura.		
Atributos	Tipo	Descripción
id_asig	bigint (19)	Etiqueta única que identifica el módulo por asignatura.
nombre_asig	varchar (255)	Almacena el nombre de la asignatura.
id_mod	bigint (19)	Etiqueta que contiene la información del módulo.

Tabla 22: Descripción del Módulo por año académico (Elaboración propia).

Año_academico		
Descripción: esta tabla contiene la descripción de los atributos del módulo por año académico.		
Atributos	Tipo	Descripción
id_año_acad	integer (10)	Etiqueta única que identifica el módulo por año académico.
num_año	integer (10)	Almacena el número del año académico.
id_mod	bigint (19)	Etiqueta que contiene la información del módulo.

Tabla 23: Descripción de la relación de Recursos Materiales y Módulo (Elaboración propia).

Rel_Mod		
Descripción: esta tabla contiene la descripción de la relación de recursos materiales y el módulo.		
Atributos	Tipo	Descripción
id_mod	bigint (19)	Etiqueta única que contiene la información del módulo.
id_rec_mat	bigint (19)	Etiqueta única que contiene la información del recurso material.

Tabla 24: Descripción de la relación de Recursos Materiales y Estudiante (Elaboración propia).

Recursos_materiales_Estudiante		
Descripción: esta tabla contiene la descripción de la relación de Recursos materiales y Estudiante.		
Atributos	Tipo	Descripción
solapin_est	varchar (255)	Etiqueta única que contiene la información del estudiante.

id_rec_mat	bigint (19)	Etiqueta única que contiene la información del recurso material.
------------	-------------	--

Tabla 25: Descripción de la relación de Recursos Materiales y Trabajador (Elaboración propia).

Trabajador_Recursos_materiales		
Descripción: esta tabla contiene la descripción de la relación de Recursos materiales y Trabajador.		
Atributos	Tipo	Descripción
solapin_trab	varchar (255)	Etiqueta única que contiene la información del trabajador.
id_rec_mat	bigint (19)	Etiqueta única que contiene la información del recurso material.

Tabla 26: Descripción de Rol (Elaboración propia).

Rol		
Descripción: esta tabla contiene la descripción de los tipos de roles que puede tomar el usuario.		
Atributos	Tipo	Descripción
id_rol	bigint (19)	Etiqueta única que contiene la información del rol.
tipo_rol	varchar (255)	Almacena el tipo de rol que ocupa el usuario.
usuario	varchar (255)	Etiqueta única que contiene la información del usuario.

Anexos E: Diseños de casos de pruebas

Tabla 27: CP2 Eliminar Libro (Elaboración propia).

Descripción general Permitir eliminar un libro en el sistema.			
Condiciones de ejecución Para poder eliminar un libro del sistema hay que: - Estar autenticado en el sistema con el rol de Encargado del almacén. -Debe existir al menos un libro en el sistema.			
SC 2 Eliminar libro			
Escenario	Descripción	Respuesta del sistema	Flujo Central
EC 2.1 Opción de eliminar libro.	Selecciona la opción de eliminar libro.	Muestra un mensaje de confirmación. Permite: -Aceptar -Cancelar	Administración/Listado de libros/Eliminar. Administración/Listado de libros/Ver/Eliminar.

EC 2.2 Opción Aceptar	Selecciona la opción Aceptar.	El sistema elimina el libro. Muestra un mensaje de información: "Libro eliminado con éxito". Regresa al listado de libros actualizado.	Administración/Listado de libros/Eliminar/Aceptar. Administración/Listado de libros/Ver/Eliminar/Aceptar.
EC 2.3 Opción Cancelar	Selecciona la opción Cancelar.	Regresa a la vista anterior.	Administración/Listado de libros/Eliminar/Cancelar. Administración/Listado de libros/Ver/Eliminar/Cancelar.

Tabla 28: CP 3 Modificar datos del libro (Elaboración propia).

Descripción general Permitir modificar los datos de un libro.			
Condiciones de ejecución Para modificar los datos de un libro hay que: -Estar autenticado en el sistema con el rol de Encargado del almacén. -Debe existir al menos una institución.			
SC 3 Modificar libro			
Escenario	Descripción	Respuesta del sistema	Flujo Central
EC 3.1 Opción de modificar los datos del libro.	Selecciona la opción de modificar los datos del libro.	Brinda la posibilidad de modificar un libro, mostrando los siguientes datos: - Título -Autor -Código - Volumen -Tomo -Cantidad Precio. Además, permitirá: Guardar -Cancelar.	Administración/Listado de libros/Editar. Administración/Listado de libros/Ver/Editar.
EC 3.2 Opción de guardar los datos	Introduce nuevos datos o modifica los ya existentes y selecciona la opción Guardar.	Valida los datos. Actualiza los datos del libro. Regresa al listado de libros y muestra un mensaje de información.	Administración/Listado de libros/Editar/Guardar. Administración/Listado de libros/Ver/Editar/Guardar.
EC 3.3 Opción de cancelar	Selecciona la opción Cancelar.	Regresa al listado de libros sin guardar los cambios realizados. Muestra un mensaje de información.	Administración/Listado de libros/Editar/Cancelar. Administración/Listado de libros/Ver/Editar/Cancelar.
EC 3.4 Campos vacíos	Existen campos vacíos.	Muestra un mensaje de información. Muestra un indicador de los campos vacíos: "(*) campos obligatorios". Regresa al EC 3.1.	Administración/Listado de libros/Editar/Guardar. Administración/Listado de libros/Ver/Editar/Guardar.

EC 3.5 Datos incorrectos	Existen datos incorrectos.	Muestra un mensaje de información. Muestra un indicador de los campos que están incorrectos. Regresa al EC 3.1.	Administración/Listado de libros/Editar/Guardar. Administración/Listado de libros/Ver/Editar/Guardar.
---------------------------------	----------------------------	---	--

Tabla 29: CP4 Ver datos de un libro (Elaboración propia).

Descripción general <i>Permitir ver los datos de un libro.</i>			
Condiciones de ejecución Para ver datos de un libro del sistema hay que: - Estar autenticado en el sistema. -Debe existir al menos un libro en el sistema.			
SC 2 Eliminar libro			
Escenario	Descripción	Respuesta del sistema	Flujo Central
EC 2.1 Opción de ver los datos de un libro	Selecciona la opción de ver los datos de un libro y consulta sus datos.	Muestra los siguientes datos del libro: -Titulo -Autor Código -Volumen -Tomo - Cantidad -Precio Permite: -Modificar los datos del libro. -Eliminar los datos del libro. -Cancelar la operación.	Administración/Listado de libros/Ver.
EC 2.2 Opción Cancelar	Selecciona la opción Cancelar.	Regresa al listado de libros. Muestra un mensaje de información.	Administración/Listado de libros/Ver/Cancelar.
EC 2.3 Opción de modificar los datos.	Selecciona la opción Editar los datos del libro.	El sistema brinda la posibilidad de modificar los datos del libro. <u>Ver CP: Modificar datos del libro.</u>	Administración/Listado de libros/Ver/Editar.
EC 2.4 Opción de Eliminar un elemento.	Selecciona la opción Eliminar libro.	El sistema brinda la posibilidad de eliminar un libro. <u>Ver CP: Eliminar libro.</u>	Administración/Listado de libros/Ver/Eliminar.